

AD-A102 928

MASSACHUSETTS INST OF TECH LEXINGTON LINCOLN LAB
TACS CENTRAL CONTROL FACILITY.(U)

F/6 17/2.1

FEB 81 S B STORCH; L E TAYLOR; S N LONDON

F19628-80-C-0002

UNCLASSIFIED

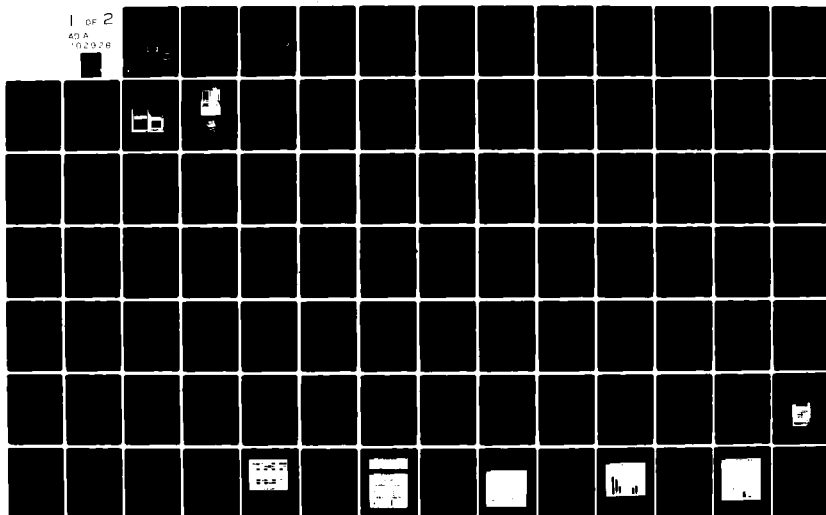
TR-542

ESD-TR-81-1

NL

1 OF 2

40 A
102928



LEVEL

12

AD A102928

Technical Report

542

TACS Central Control Facility**DTIC**

AUG 17 1981

S.B. Storch
L.E. Taylor
S.N. Landon
C.D. Cappello

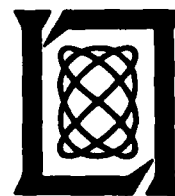
12 February 1981

Prepared for the Department of the Navy
under Electronic Systems Division Contract F19628-80-C-0002 by

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LEXINGTON, MASSACHUSETTS



Approved for public release; distribution unlimited.

81 8 17 005

DTIC FILE COPY

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. The work was sponsored by the Department of the Navy under Air Force Contract F19628-80-C-0002.

This report may be reproduced to satisfy needs of U.S. Government agencies.

The views and conclusions contained in this document are those of the contractor and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the United States Government.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Raymond L. Loiselle

Raymond L. Loiselle, Lt. Col., USAF
Chief, ESD Lincoln Laboratory Project Office

Non-Lincoln Recipients

PLEASE DO NOT RETURN

Permission is given to destroy this document
when it is no longer needed.

12

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

TACS CENTRAL CONTROL FACILITY

Steven B. STORCH
Lee E. TAYLOR
Susan N. LANDON
Carole D. CAPPELLO
Group 66

14, T-1-12

DTIC
AUG 17 1981
H

7. TECHNICAL REPORT 542

11 12 FEBRUARY 1981

Approved for public release; distribution unlimited.

LEXINGTON

MASSACHUSETTS

ABSTRACT

The Terminal Access Control System (TACS) is designed to make efficient usage of 25-kHz-wide satellite transponder channels, such as those available on FLEETSAT, GAPSAT, and LEASAT. It does so by applying time division multiple access and automated demand assignment techniques to these channels, allowing a significant increase in both data throughput and supportable user population, as compared to the initial, single network per transponder FLEETSAT operating mode. The Terminal Technology Group of Lincoln Laboratory Division 6 has built prototypes of all of the TACS ground segment equipment, as well as special-purpose test equipment used to demonstrate and quantify the performance of TACS. This report describes the implementation details of components of the TACS Central Control Facility as configured for testing at the Laboratory. The components described are: the Multiple Access Controller, the Call Simulator, the System Operator's Console, and the Scenario Generator. All differences in implementation/configuration between the testing environment and a full-scale operational environment are noted in the report.

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Availability	
A	

CONTENTS

Abstract	iii
List of Illustrations	vii
List of Tables	ix
PREFACE	xi
A. System Highlights	xi
B. System Documentation	xiv
I. INTRODUCTION	1
II. MULTIPLE ACCESS CONTROLLER	5
A. Interfaces and Data Flow	6
1. Configuration in a Full-Scale Operational Environment	6
2. Configuration in the Lincoln Laboratory TACS Central Control Facility	6
3. System Management Data Flow	7
B. Hardware Operating Environment	9
1. Computer	9
2. TACS Interfaces	9
3. Other Central Control Facility Units	10
4. Operating System Peripherals	10
C. Software Operating Environment	10
D. Real-time Structure	11
1. Considerations Influencing Real-time Structure Design	11
2. Description of Real-time Structure	12
E. Processing Structure	16
1. Considerations Influencing Processing Structure Design	16
2. Description of Processing Structure	17
F. Input Algorithm	22
G. Assignment Algorithm	27
1. Assignment Algorithm Overview	27
2. Assignment/Blockage Decision Process: All-Member Nets	31
3. Assignment/Blockage Decision Process: Point-to-Point Nets	33
III. CALL SIMULATOR	43
A. Functional Description	44
1. Overview	44
2. Realism of Simulation	44
3. Input Requirements	47
4. Output Requirements	48
5. Limitations	54
6. Conclusion	57
B. Implementation Details	57
1. Hardware	57
2. System Software	58
3. Simulation Software	58
C. Conclusion	70

IV. SYSTEM OPERATOR'S CONSOLE	71
A. Introduction	71
B. The Intecolor 8051 Intelligent Terminal	72
C. Software Design Philosophy	72
1. Interface to the Multiple Access Controller	73
2. Choice of Programming Language	73
3. Program Structure Considerations	73
D. TACS Display and Command Package	74
1. Frame Structure Display	75
2. Request Queue Display	77
3. System Statistics Display	79
4. Detailed Slot Data Display	81
5. Satellite Capacity Utilization Display	83
6. Call Precedence Distribution Display	83
7. Command Mode	83
E. Detailed Software Description	87
1. Overall Program Design	87
2. Program Initialization	87
3. Receiving Data from the Multiple Access Controller	87
4. Frame Structure Display	89
5. Request Queue Display	91
6. System Statistics Display	93
7. Detailed Slot Data Display	95
8. Satellite Capacity Utilization Display	99
9. Call Precedence Distribution Display	101
10. Polling the Keyboard for Operator Signalling	101
11. Command Mode	103
F. Display Generation on the Intecolor 8051 Color Terminal	105
G. Input/Output Handling on the Intecolor 8051 Terminal	105
H. Program Development Information	107
V. SCENARIO GENERATOR	109
A. Motivation for Automated Scenario Generation	109
B. Description of Scenario Generator	110
Acknowledgments	114
References	115
Glossary	116
Appendix A FRAME STRUCTURE DESIGN	125
Appendix B (EXAMPLE OF) CIRCUIT ESTABLISHMENT/TERMINATION TIMING	131
Appendix C SYSTEM MANAGEMENT DATA FORMATS	135
Appendix D FIVE- AND NINE-SLOT SYSTEM IMPLEMENTATION DIFFERENCES	147
Appendix E MULTIPLE ACCESS CONTROLLER DATA BASE	153
Appendix F VARIAN MINICOMPUTER MEMORY MAP	159
Appendix G MAC ALGORITHM FOR DETERMINATION OF LOCAL TDMA SLOT AVAILABILITY	163
Appendix H VARIAN CALL SIMULATOR	171

LIST OF ILLUSTRATIONS

P-1	System Configuration	xii
P-2	Frame Structure (including system control circuits)	xii
1-1	Lincoln Laboratory TACS Central Control Facility	2
1-2	Multiple Access Controller and System Operator's Console	2
1-3	Master TAC	3
1-4	Call Simulator	3
2-1	MAC Hardware Operating Environment	8
2-2	MAC Real-time Structure	13
2-3	MAC Processing Structure	18
2-4	Primary Inputs and Outputs for MAC Processing Routines	19
2-5	Input Algorithm (IALG) Flowchart	23
2-6	Assignment Algorithm (AALG) Flowchart (Emphasizing request queue and control burst management)	26
2-7	Call Progress Messages	29
2-8	Flowchart of Assignment/Blockage Decision Process for All-Member Net Requests	30
2-9	Example of All-Member Net Assignment Algorithm	33
2-10	Flowchart of Assignment/Blockage Decision Process for Point-to-Point Net Requests	37
3-1	Call Simulator Software/Buffering Structure	46
3-2	Call Simulator Hardware Operating Environment	56
3-3	Call Simulator Real-time Structure	60
3-4	Overview of Call Simulator Processing Structure	61
3-5	Flowchart of Error Checking Done on Control Burst Receipt	62
3-6	Flowchart of Call Simulator Routines for Updating Data Base	64
3-7	Request/Report Generation, Holding and Sending Routines	65
3-8	Request Parameter Generation Routines	67
4-1	Photograph of Intecolor 8051 Terminal as TACS System Operator's Console	71
4-2	Display Menu	75
4-3	Frame Structure Display for the Nine-slot Frame	76
4-4	MAC Request Queue Display	78
4-5	System Statistics Display	78
4-6	Prompting Sequence for Detailed Slot Data Display	80
4-7	Detailed Slot Data Display	80
4-8	Satellite Capacity Utilization Bar Graph	82
4-9	Call Precedence Distribution Bar Graph	84
4-10	System Operator's Console Program Design	86
4-11	Frame Structure Display Flowchart	88
4-12	Request Queue Display Flowchart	92
4-13	System Statistics Display Flowchart	92
4-14	Detailed Slot Data Display Flowchart	96
4-15	Satellite Capacity Utilization Bar Graph Flowchart	98
4-16	Call Precedence Distribution Bar Graph Flowchart	102

LIST OF ILLUSTRATIONS (Continued)

4-17 Keyboard Polling Flowchart	102
4-18 Command Mode Flowchart	106
5-1 Scenario Generator Flowchart	111
A-1 Nine-Slot Frame Structure	129
A-2 Five-Slot Frame Structure	129
B-1 Timing of the Establishment and Termination of a Three Frame Assignment	132
C-1 General Control Burst Data Format	136
C-2 Control Burst Message Format	138
C-3 Status Report, Request, and Ranging Burst Formats	140
C-4 System Status Data Stream	144
C-5 System Status Data Stream Request Queue Block Format	144
C-6 System Operator Command Format	144
D-1 Frame Structure Display for the Five-slot Frame	150
E-1 Format of Request Queue Block and Assignment List Block	154
E-2 Assignment List Codes for Slots Not Currently Assigned to a Net	156
E-3 Format of TAC List Block	156
E-4 Format of Net List Block	158
E-5 Format of Segment of Satellite Capacity Maps	160
F-1 Varian Minicomputer Memory Map	162
G-1 Transceiver Configuration/Activity Summary Table (generated by XSUM)	164
G-2 Frame Structure Timing Relationships	167
G-3 Example Frame Timing Diagrams	167
G-4 Example Transceiver Configuration/Activity Summary Table	168
G-5 Inequalities and Slot Maps For Example Half-Duplex Platform	169
G-6 Inequalities and Slot Maps For Example Full-Duplex Platform	170

LIST OF TABLES

3-1	Request Parameters	49
A-1	Candidate Frame Durations	126
A-2	Specifications Influencing Frame Format for Use With 32 kbps Modem	126
A-3	Data and System Control Burst Durations and Packing Efficiencies (Nine-Slot Frame)	128
A-4	Specifications Influencing Frame Format for Use with 19.2 kbps Modem	130
A-5	Data and System Control Burst Durations and Packing Efficiencies (Five-Slot Frame)	130
D-1	Burst/Code Rate Combinations, and Resulting Number of Contiguous Slots for Assignments in Five- and Nine-Slot System	148

PREFACE

The Terminal Access Control System is designed to allow efficient usage of 25-kHz-wide satellite transponder channels, such as those available on FLEETSAT, GAPSAT and LEASAT. By applying time division multiple access and real-time (i.e., automatic) demand assignment techniques to these channels, TACS significantly increases both data throughput and supportable user population, when compared to the initial, single network per transponder FLEETSAT operating mode. TACS employs centralized demand assignment control, and prioritizes all message traffic so that equitable preemption is possible when important calls find the system fully occupied.*

In order to demonstrate the feasibility and quantify the performance of TACS, the Terminal Technology Group of Lincoln Laboratory Division 6 has built prototypes of all the equipment comprising the TACS ground segment. Units were built in sufficient numbers that two fully-equipped TACS communications sites could be set up: a remote subscriber station and the Central Control Facility. The Central Control Facility includes the Call Simulator (a piece of special test equipment) which creates the appearance that there exists a large active population of remote terminals. This simulator allows testing of TACS demand assignment performance in a full-scale deployment scenario. Before commencing with this report's description of the implementation of the Central Control Facility, an overview of the system is provided.

A. System Highlights

The TACS equipment configuration is depicted in Fig. P-1. A remote station ("Mobile Platform") installation consists of an existing UHF transceiver such as the half-duplex WSC-3 and a TACS subscriber unit (TAC) which supports up to four user I/O devices. The TAC's major functions are to perform synchronization to the system time base, multiplexing/demultiplexing, error control encoding/decoding and circuit requesting/relinquishment. Although only two are shown, there might be several hundred remote stations. One active central control site, with backups for survivability, is required. The central control site installation consists of a bank of TACs and transceivers (preferably full duplex and one per channel being managed by TACS), many user devices, and the central demand assignment control minicomputer (MAC) facility,

* The Navy has contracted with Motorola Corporation's Government Electronics Division for development of the "Semiautomatic" UHF DAMA System, which incorporates a number of the TACS design features. That system is semi-automatic in that circuit assignment is done by a human operator rather than a computer at the Master Control Station. The Semiautomatic DAMA terminal hardware, however, has been specified such that it can operate in a fully automated mode; thus the intended upgrading to automatic assignment will only require the addition of the MCS computer. Semiautomatic DAMA terminal production is scheduled to start at the beginning of FY 81, with MCS upgrading anticipated after a few years of deployment. At that point, it is presumed that most of the TACS capabilities will be placed in operational usage.

105501-S

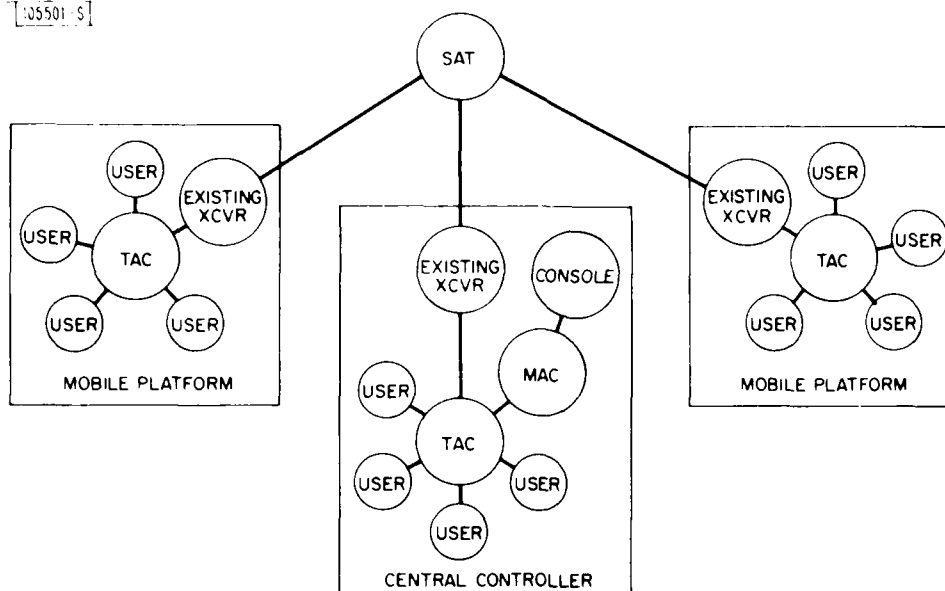
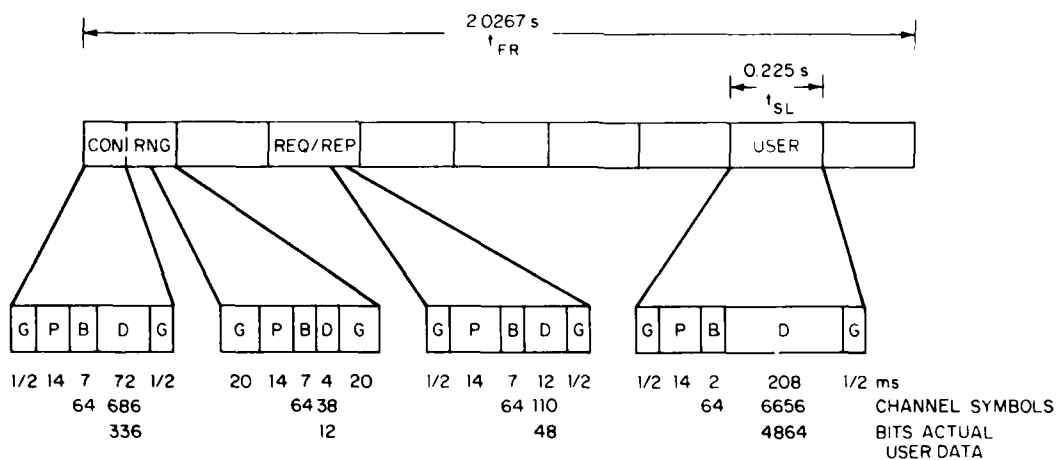


Fig. P-1. System configuration.



CON: CONTROL SUBSLOT
RNG: RANGING SUBSLOTS (2)
REQ/REP: REQUEST AND STATUS REPORT SUBSLOTS (6)
USER: 2400 baud USER DATA SLOT
G: GUARD BAND
P: MODEM ACQUISITION PREAMBLE
B: BURST SYNCHRONIZATION SEQUENCE
D: DATA

105502-N

Fig. P-2. Frame structure (including system control circuits).

which includes a System Operator's Console. The MAC maintains a data base which contains all the relevant information on the configurations and status of all terminals in the system, as well as the satellite channels. Using the information in the data base, together with circuit requests, ranging signals and reports, the MAC performs its three main functions: system time base management, circuit assignment, and adaptive control of system parameters.

Although TACS is capable of supporting typical point-to-point communications with total connectivity, the system is normally configured in a way more applicable to netted military operations. TACS can support data nets of two distinct types, designated "all-member" and "point-to-point." An all-member net corresponds to the traditional military network, its more important characteristics being that when the net is on the air, all members are expected to continuously monitor net traffic and to respond if polled by a net controller. All-member nets traditionally have large populations (i.e., tens of members). The point-to-point net more closely resembles the commercial telephone system, and as implied by its name, it is designed for two-party calls. TACS allows multipoint (conference) calls within a point-to-point net. Additional parties are added one at a time, via a new circuit request for each additional party.

The TACS central controller synchronizes all subscriber units to time frames of duration approximately 2 s. The frame is divided uniformly into nine basic data slots, each of which could accommodate an independent 2400-bps data circuit among terminals with nominal link quality. Figure P-2 shows the frame structure, for a channel containing the system control circuits. The control burst (generated by the MAC) contains a pseudonoise sequence from which remote subscriber units can derive receiver (i.e., downlink) frame sync and also includes control messages directed to the remote users. The pair of ranging subslots, which are monitored by the MAC, is provided so that remote units can send short bursts to the MAC, which can use the burst arrival times to compute transmitter (i.e., uplink) frame sync commands. Finally, the six request and status report subslots allow the remote users to inform the MAC of their current status and immediate needs for communications circuits. Request subslots are continuously available to all terminals, under a shared or random-access protocol. In addition, dedicated usage of the report subslots, which have sufficient capacity to carry both a report and a request, cycles through the user community via a round-robin protocol.

TACS is highly automated, requiring very little human operator control and a minimum of user/system interaction. At the central control site the system operator's presence is required only when it is necessary to add or delete channels from the pool under TACS control. The equipment operator of a remote site is needed only to initially configure the subscriber unit - e.g., to specify which net is to use which I/O port. Finally, the user interfaces with the demand assignment features of TACS via a simple device named the Terminal Input Controller (TIC), consisting of a keyboard number pad and a one-

line alphanumeric display. The keyboard allows the user to enter call parameters, such as message precedence, and to signal for hangup. The display informs the user of the status of any pending circuit requests and of any impending call preemptions. In the background as far as human intervention is concerned, the TACs and MAC automatically handle: terminal synchronization, link status reporting/link adapting, circuit assignment, and system control circuit configuration (e.g., the number of request slots varies with time as a function of measured request rate).

B. System Documentation

Quite a volume of preliminary documentation was written during the early phases of TACS ^{1,2,3}. This report is one of three constituting the final TACS documentation package. Of the other two reports, Taylor and Bernstein⁴ presents the top-level system design and the performance test results. Landon et al.⁵ describes implementation details for the subscriber units.

This report provides details of the implementation of the TACS Central Control Facility. Reader knowledge of TACS concepts and of the higher-level elements of TACS design is presumed. That material is most readily available in Reference 4.

I. INTRODUCTION

A subset of the components of the Terminal Access Control System (TACS) as implemented at Lincoln Laboratory is known as the "Central Control Facility." The facility consists of the Multiple Access Controller (MAC), the Call Simulator, the System Operator's Console, and the "Master" Terminal Access Controller (Master TAC). Figure 1-1 is a block diagram showing the configuration of the Central Control Facility. The types of data transferred between the four units are indicated. Figure 1-2 is a photograph of the MAC and the System Operator's Console, Fig. 1-3 depicts the Master TAC, and Fig. 1-4 depicts the Call Simulator.

This report is organized around a unit-by-unit description of the Central Control Facility. Sections to follow describe the MAC, the Call Simulator, and the System Operator's Console. As the implementation details of the Master TAC unit are similar to those of a "remote" TAC unit, they are presented in Landon et al.⁵ rather than in this document. An additional section has been included in this document to describe the "Scenario Generator," a user-interactive computer program greatly simplifying the construction of TACS operating environments. A number of appendices and a glossary are also included with the report. The appendices provide further implementation details, many of which will assist the reader in understanding the material in the body of the report. The glossary should assist the reader by clearly defining the many terms and acronyms that (necessarily) have come to be used in descriptions and discussions of TACS.

It should be noted that the MAC, Call Simulator, System Operator's Console, and Scenario Generator software described in this report has been designed to work with the nine-slot frame structure described in Appendix A. This frame structure requires all subscriber units to be equipped with 32 kbps modems. As the TAC units built at Lincoln Laboratory currently operate at 19.2 kbps, a five-slot frame structure (also described in Appendix A) has been designed for system operations with scenarios containing real as well as simulated TACs. This requires additional, five-slot frame versions of MAC, Call Simulator, and Scenario Generator software. (A single version of the System Operator's Console software can accommodate either five- or nine-slot system operations; see Section IV.) Unless specific reference is made to the five-slot frame structure, all implementation details presented in this report pertain to the nine-slot frame. The differences between the five- and nine-slot systems are enumerated in Appendix D.

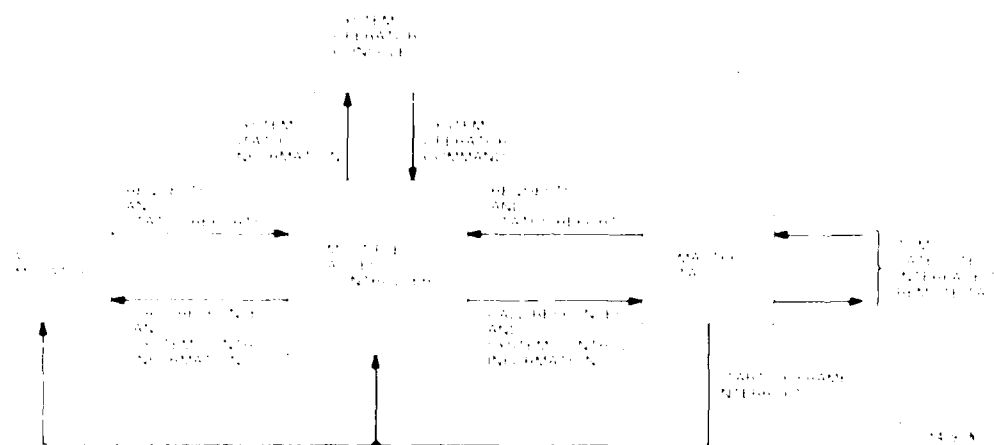


Fig. 10. Time-dispersed laboratory TACs central control facility.

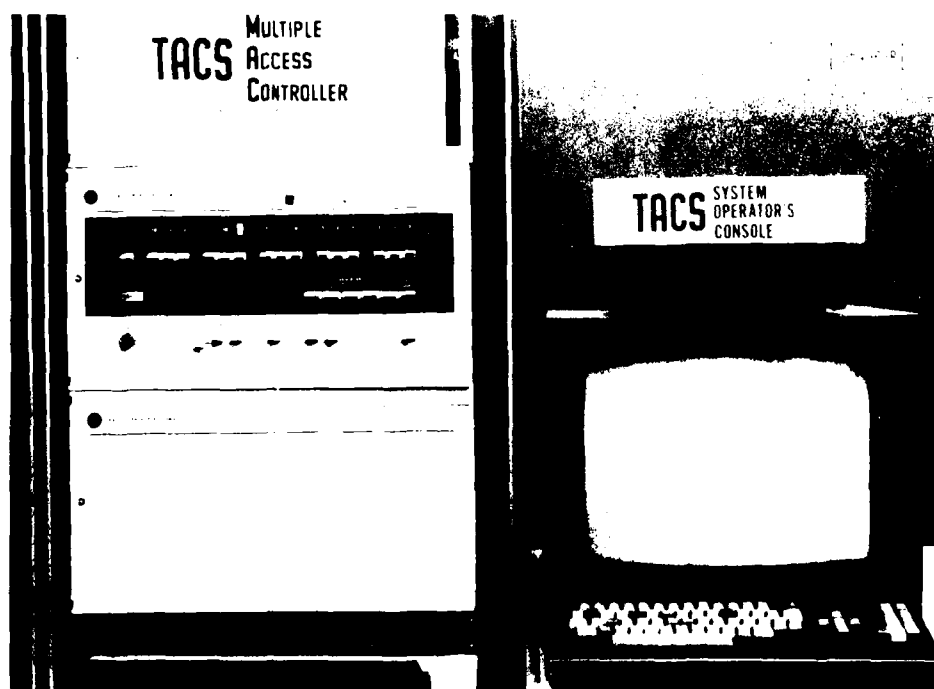


Fig. 1-2. Multiple access controller and systems architecture.



Fig. 1-3. Master TAC.



Fig. 1-4. Call simulator.

II. MULTIPLE ACCESS CONTROLLER

The Multiple Access Controller (or MAC) is implemented on a minicomputer and located at the Central Control Facility. Its jobs are to provide remote terminals with uplink time synchronization commands and to control user access to the satellite transponder channels.

Remote terminals send initial ranging probes (with ambiguity of ± 20 ms) and range update probes (with ambiguity ± 0.5 ms) to the central control site via special time slots in one of the satellite transponder channels. A subscriber unit at the central site measures arrival time of the probes and passes that information on to the MAC. The MAC's part in the uplink sync process is to calculate the timing offset (Δt) from nominal, to figure out the address of the probing terminal, and to send a time slew command to that terminal.

By far the larger task of the MAC is managing usage of the satellite resources. Requests for circuits from remote terminals are received by a subscriber unit at the central control site and passed to the MAC, which initially just places them in a queue. Accessing the request queue in order of message precedence and time of arrival, the MAC considers requests one by one. Requests normally receive a response, usually an assignment of capacity but sometimes a "busy" message, within two time frames. Consideration of a request is a two stage process. In stage one, the MAC references its extensive data base to determine the present status of all terminals which are parties to the call. The critical status items are the individual terminal's link condition and any ongoing data traffic. Poorest link quality among the terminals which are parties to the call determines code and symbol rates, and thus indirectly the number of slots which are needed to support the call. Ongoing traffic at some of the terminals may restrict the location (i.e., which slots of the nine per time frame) of the slot(s) used for the call. In stage two, the MAC takes as input the near-optimum set of slots for the call, and determines if those slots are available (idle) in any of the transponder channels. The first available slots found are assigned to the call. If the first iteration of request consideration turns up no available capacity, the MAC reiterates, allowing for preemption of lower precedence ongoing calls. The end result of the iterative process is one of two decisions: to assign capacity for the call or to reject the request. In either event, the MAC composes and transmits a response message, clears the request under consideration from the queue, and proceeds to try to honor the next queued request, if any. Calls which are assigned capacity leave the system in one of three ways: preemption by the MAC (rare), expiration of the time requirement (if initially specified in the request), or upon MAC receipt of a hangup signal from the requesting party.

The MAC applications software, which was not optimized (for either run time or memory requirements), occupies 13.5K words of 16-bit memory; the program itself uses 4.5K words, with tables and buffers using the other 9K words.

Measurements revealed that in the worst-observed case, the MAC minicomputer was only actively processing at 0.1 duty cycle, the rest of the time available for processing being spent in "wait loops." Thus the mid-1960's technology, 32K word minicomputer was more than adequate for MAC implementation. Future similar, demand assignment systems could very well use microprocessor-based central controllers.

A. Interfaces and Data Flow

1. Configuration in a Full-Scale Operational Environment

The MAC may be viewed as the central control node of TACS. It must interface with every subscriber unit in the TACS user community. Communication between the MAC and remote subscriber units occurs over the satellite links via specially designated subslots (the control, request/report, and ranging subslots) provided in the Time Division Multiple Access (TDMA) frame. Since the MAC has no direct satellite interface, the transceiver(s) belonging to a (number of) TAC unit(s) located on the same platform or ground station as the MAC is (are) used to link the MAC to remotely located subscriber units (see Fig. P-1). In an operational Naval system this Central Control Facility would probably be located at Communications Area Master Station (CAMS) or a COMMSTA. An array of up to nine TAC units and associated transceivers, ideally though not necessarily one for each satellite channel under TACS control, would be located at the Central Control Facility. Data are passed between the MAC and the transceiver (array) via a hard-wired interface. The MAC must also interface with the system operator. This is accomplished by another hard-wired link to the System Operator's Console. Thus, every system management data transfer in TACS is routed through the MAC. Note that the MAC has no interface with the data being passed between the TACS end users via assigned circuits; the MAC plays no role in circuit management other than circuit assignment and termination.

2. Configuration in the Lincoln Laboratory TACS Central Control Facility

The Lincoln Laboratory TACS test configuration (as contrasted to an operational TACS configuration) contains only two subscriber units. One of these is designated as the Master TAC: It accesses satellite subslots, allowing (control burst, request, status report, and ranging burst) data transfer between the MAC and the remotely located subscriber unit (the "remote" TAC). System management data generated locally by users at the Master TAC are sent directly to the MAC, along with the system management data received from the remote TAC. The Master TAC also provides the MAC with a time base for real-time operations by sending an interrupt pulse to the MAC at the start of every time frame.

Another component of the TACS test configuration is the Call Simulator, a minicomputer-based simulation of a large community of TACS subscriber units. The Call Simulator interfaces with the MAC in the same fashion as the Master TAC: The system management data passed between the MAC and Call Simulator are

virtually identical to the data which would be passed (in a full-scale system) between the MAC and the TAC units at the central control site in order to service a large TACS user community. This arrangement requires MAC management of an interface (the Call Simulator interface) that wouldn't exist in a full-scale operational system. It also requires that the MAC merge the system management data from the real and simulated TAC units. These disadvantages are far outweighed by the power and flexibility in all phases of MAC verification and testing that are obtained through the use of the Call Simulator.

Yet another Lincoln Laboratory TACS component communicating directly with the MAC is the System Operator's Console. This unit, the Master TAC, the Call Simulator, and the MAC itself, comprise the Lincoln Laboratory TACS Central Control Facility. The block diagram of the Central Control Facility (Fig. 1-1) graphically depicts the MAC's role as the controlling node of the facility, and, therefore, of the entire TAC system. The diagram indicates the type of data passed between the MAC and each of the three units with which it must interface.

3. System Management Data Flow

Every frame, the MAC accepts system management data from each of the other three Central Control Facility units. It then merges and processes the data, making the appropriate changes in its data base. Finally, system management data are sent out to each of the units reflecting the results of the MAC's processing. All of this must occur within an amount of time equivalent to one time frame's duration (approximately 2s). The time base for these real-time operations is provided by the Master TAC's start of frame (SOF) interrupt pulse.

The input from both the Master TAC and the Call Simulator consists primarily of requests and status reports. The requests contain parameters, supplied by the end users located at the subscriber units' I/O ports, that are required by the MAC's circuit assignment algorithm. Status reports contain information on the current communications activity, satellite range, and link quality of the subscriber unit making the report. In addition to request and status report input, ranging bursts are sent to the MAC (via the Master TAC) by the Remote TAC when it first enters the system. The input from the System Operator's Console consists of commands, generated by the system operator, all of which alter the current TACS communication configuration.

The output system management data sent to both the Master TAC and the Call Simulator (i.e., the control burst data) consist of both call (request) responses such as circuit assignments and "busy signals," and system control information such as preemptions of circuits and satellite range updates. The System Operator's Console receives data, derived from the MAC's data base, indicating overall system activity. This information is used to generate various displays of current system status.

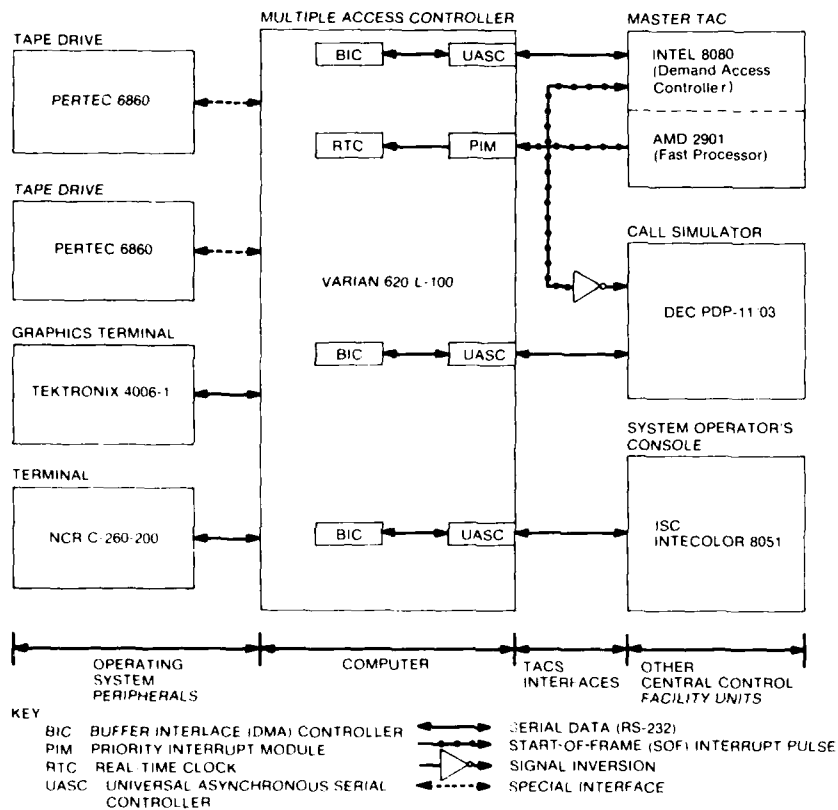


Fig. 2-1. MAC hardware operating environment.

Two particular aspects of TACS design heavily impact the I/O and processing functions of the MAC. The first is the finite amount of system management data capacity in the TACS TDMA frame. Reasonable system response time and stability require that the MAC make use of as much of this capacity as possible. Therefore, not only must the MAC perform I/O operations with all Central Control Facility units in every frame, but it must process as much of the input data as possible, in an ordered fashion, until it has either exhausted the output data capacity (control burst capacity) or the processing time allotted for the frame. The second design aspect is the requirement that the MAC's data base always truly reflect the on-going communications activity in the (real and/or simulated) TACS user community. This implies that (1) status reports from subscriber units must be processed as quickly as possible, and (2) the MAC must not make any changes in its data base that aren't immediately forwarded to the user community in the control burst. These issues will be discussed in greater detail in subsequent sections.

B. Hardware Operating Environment

Figure 2-1 is a diagram of the hardware implementation of the MAC and its interfaces with the other Central Control Facility units. The various items shown are divided into four categories: (1) computer (hosting the MAC), (2) TACS interfaces, (3) other Central Control Facility units, and (4) operating system peripherals. Each category is discussed below.

1. Computer

The MAC has been implemented on a Varian 620/L-100 minicomputer containing 32K 16-bit words of core memory with a 950-ns cycle time. Front-panel sense switches are available to control program flow. The following CPU options are required to run the MAC program: extended addressing, hardware multiply/divide, and interrupt. A Varian Real-Time Clock (RTC) module is also required by the MAC program to control its internal real-time events. The real-time clock must be set to operate at 1 kHz.

2. TACS Interfaces

The MAC exchanges data with the other TACS Central Control Facility units via three Varian "Universal Asynchronous Serial Controller" (UASC) cards. The cards implement an RS-232 standard interface. All controllers are set to operate at a data rate of 9600 baud. These serial ports are used by the MAC in a direct memory access (DMA) mode, requiring three "Buffer Interlace Controller" (BIC) cards, one to be associated with each UASC card.

One further TACS interface is required: an interface to the start of frame (SOF) interrupt pulse forwarded to the MAC by the Master TAC. This is provided by a Varian "Priority Interrupt Module" (PIM) card. The start of frame interrupt supplied via the PIM is the principal time base for the MAC's real-time operations. The Real-Time Clock module is used to derive internal MAC real-time

events based on this SOF interrupt. This relationship is illustrated in Fig. 2-1. The figure also indicates the receipt of the SOF interrupt pulse by the Call Simulator, after a stage of inversion. The inversion is required due to the opposing pulse-edge triggering requirements of the PIM in the Varian and the Call Simulator's interrupt interface.

3. Other Central Control Facility Units

The Call Simulator has been implemented on a Digital Equipment Corporation (DEC) PDP-11/03 minicomputer. The PDP-11/03 contains a programmable real-time clock and an external interrupt interface. I/O from/to the MAC is handled via DEC programmed I/O rather than direct memory access. Section III contains further Call Simulator implementation details.

The Master TAC consists of two parts: (1) the "Fast Processor," implemented on an Advanced Micro Devices (AMD) 2901 microprocessor, and (2) the "Demand Access Controller" (DAC), implemented on an Intel 8080 microprocessor. The DAC is the part of the Master TAC communicating with the MAC. DAC I/O from/to the MAC is handled via direct memory access. The DAC does not have a programmable real-time clock. The Fast Processor generates the SOF interrupt pulse, which is passed to the DAC as well as to the MAC and Call Simulator. For Master TAC implementation details, see Landon et al.⁵.

The System Operator's Console is an Intelligent Systems Corporation (ISC) Intecolor 8051 intelligent color graphics terminal. Communication with the MAC is done at the console via programmed I/O. The Operator's Console does not have an interrupt module or a programmable real-time clock, and therefore requires a special real-time software interface with the MAC. The System Operator's Console is described in further detail in Section IV.

4. Operating System Peripherals

A number of the items shown in Fig. 2-1 as being interfaced to the Varian minicomputer are not integral parts of the TACS Central Control Facility. The Tektronix 4006-1 graphics terminal, the National Cash Register (NCR) C-260-200 terminal, the two Pertec 6860 tape drives, and all associated controllers are used by the operating system resident in the Varian. This operating system, described in the next section, provided the software environment necessary to develop and test the MAC. Although a few MAC routines do directly address the operating system peripherals, such use is only for initialization, verification, and testing of the MAC's operation in TACS.

C. Software Operating Environment

The MAC was developed and tested under the control of a magnetic tape operating system developed at Lincoln Laboratory. This operating system, known as the "Sanguine Simulation Facility" ⁶ or "Debug," allows transfer of programs and data between Varian memory and magnetic tape, examination/modification of register or memory contents, setting of program break points, etc. All other

necessary program development tools (editor, assembler, tape file manipulation programs, etc.) are available under the control of the operating system.

All MAC code was written in Varian assembly language. The assembler that was used was originally supplied by Varian, but was modified extensively at Lincoln Laboratory. The high-speed line printers at the Lincoln Laboratory IBM 370/168 facility were used to generate program listings.

Documentation of all MAC support software is available.⁶ Appendix F contains a map showing the organization of the MAC software and the Debug operating system software in the Varian minicomputer's memory.

D. Real-time Structure

1. Considerations Influencing Real-time Structure Design

I/O considerations have necessarily had a great effect on the design of the MAC's real-time structure. The three RS-232 standard serial interfaces connecting the Varian to the other Central Control Facility units must be serviced bi-directionally during every frame. Servicing these serial ports via direct memory access (DMA) I/O provides a number of advantages over programmed I/O operations: (1) DMA allows for more efficient use of Varian CPU time, requiring only a trivial number of machine instructions to initiate I/O operations. (2) The servicing of a number of simultaneously operating asynchronous interfaces is automatically handled by the DMA controllers, simplifying the required I/O-handling code. The controllers transfer data to computer memory only when and if such data have been received. This feature allows the MAC to service only those interfaces that are active, i.e., the MAC will operate whether or not all of the other Central Control Facility units are connected to it.

The quantity and type of data passing through each MAC interface in a frame have been defined, for the most part, by system-level design parameters. For a detailed description of these data, see Appendix C: "System Management Data Formats." The minimum amount of time (per frame) that each interface transfers data is therefore fixed, given the data rate (9600 baud) and the Varian hardware limitation of doing DMA I/O in only one direction at a time for a port. A number of factors have influenced the locations in the frame (relative to the SOF interrupt) at which these transfers occur. One such factor is the capabilities of the other Central Control Facility units. For example, the Demand Access Controller section of the Master TAC does not have a programmable real-time clock, and therefore passes request, status report, and ranging data to the MAC immediately after receiving the SOF interrupt pulse. As another example, the System Operator's Console, having neither a clock nor an interrupt capability, uses the data received from the MAC to determine time, sending data to the MAC immediately after transfer of the system status data stream is completed. Another factor, clearly, is the relationship of the data transfers to the MAC's processing tasks. This relationship can be very complex, given

the Varian's ability (via DMA) to perform processing tasks while I/O is in progress, and the need for the MAC to maintain as high a level of throughput as the TACS TDMA frame structure allows.

The real-time structure of the MAC has also been influenced by the need for a fairly straightforward means of verifying and testing MAC software. Since detailed examination of the MAC's status cannot occur in the TACS real-time environment, the ability to start and stop the MAC has been incorporated into the real-time structure. By the use of a Varian front-panel sense switch, MAC real-time operations are halted and the Debug operating system environment is entered. The entire TAC system is in a consistent state when the MAC is halted: the MAC, Master and Remote TACs, Call Simulator, and System Operator's Console all are in agreement as to current system communications activity. The contents of I/O buffers, data bases, etc., can be easily examined at this point. Real-time operations can then be continued without re-initialization of the run.

2. Description of Real-time Structure

Figure 2-2 is a combined flowchart and timing diagram for the MAC's real-time structure. The axis running across the center of the figure represents a single 2.027-s TACS TDMA frame. The numbers above the axis represent the time (in seconds relative to the start of frame) of events occurring during the frame. These events, designated by RS0, R0, R1, RS2, etc., are flowcharted below their positions on the frame time axis. Each event is connected to the next event by the occurrence of a real-time clock (RTC) or start of frame (SOF) interrupt, as indicated by the dashed lines. RINT designates the code segment that sets up real-time operations; hence RINT does not occur at any special time in the frame. Note that the designations RINT, RS0, R0, etc., are identical to the labels on the sections of the MAC's real-time control software handling the corresponding events. The time intervals during which the input and output from each Central Control Facility unit are expected to occur are indicated above the axis. Note that these time intervals are generally much shorter than the time intervals during which the interfaces are enabled. This provides extra time margin that may be required if input from any of the units arrives at the MAC slightly delayed. Also indicated above the axis is the time interval in the frame that is allotted for "MAC processing tasks," that is, for all MAC operations other than I/O. The initiation of the MAC processing tasks is also shown in the flowcharts for events RS2 and R2.

It can be seen from the diagram that the MAC's time base originates with the SOF interrupt, which is used to reset the real-time clock at the start of every frame. The RTC interrupt provides the timing for events within the frame. Hence, one of these two interrupts is enabled at every point in the frame. It is also of note that the I/O and processing tasks do not overlap, which is to say that the Varian's DMA I/O capability discussed in the preceding section is not being fully exploited. Testing has shown that the processing time required by the MAC is generally much smaller than (and is typically $\leq 10\%$ of)

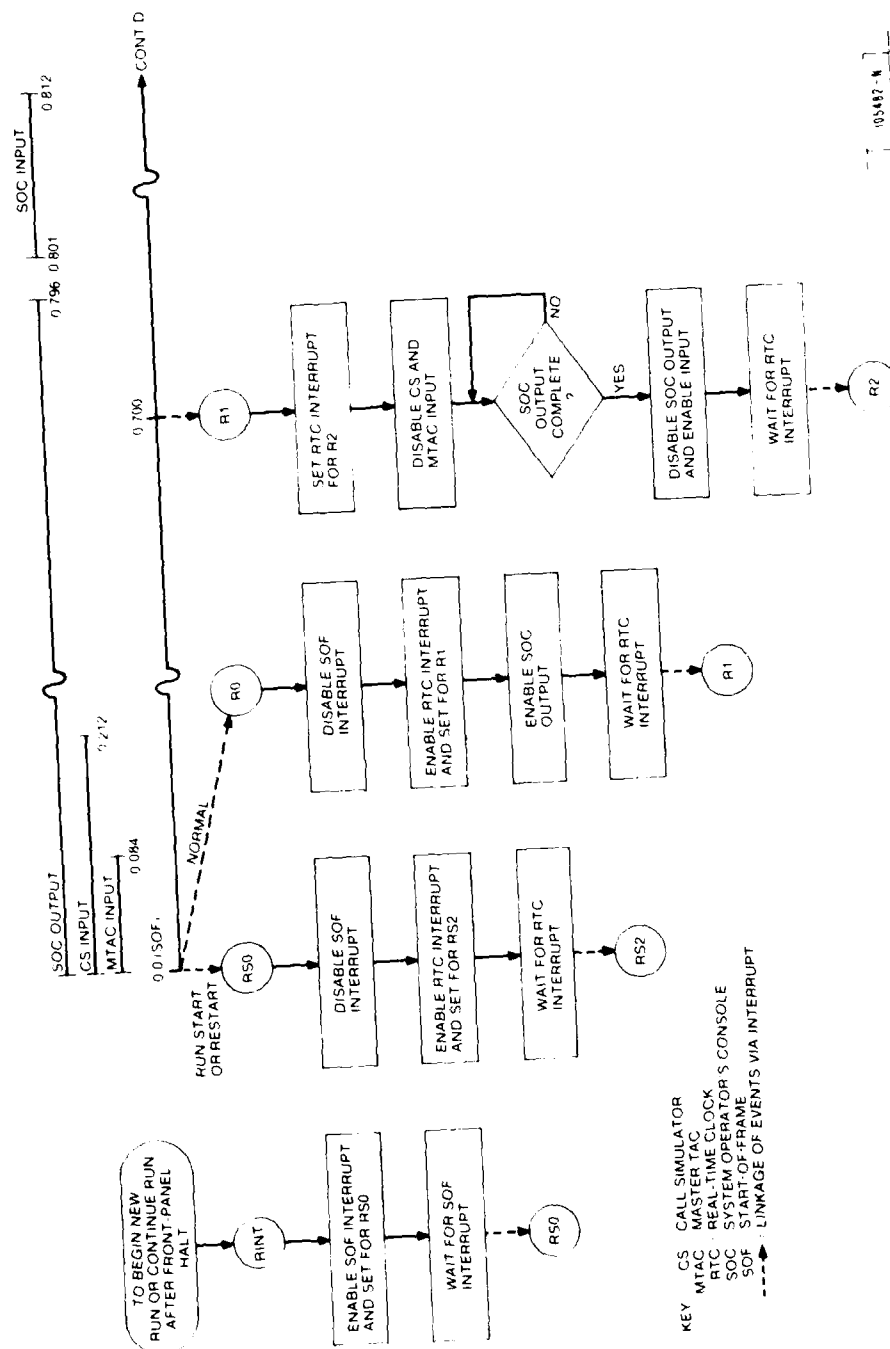


Fig. 2-2. MAC real-time structure.

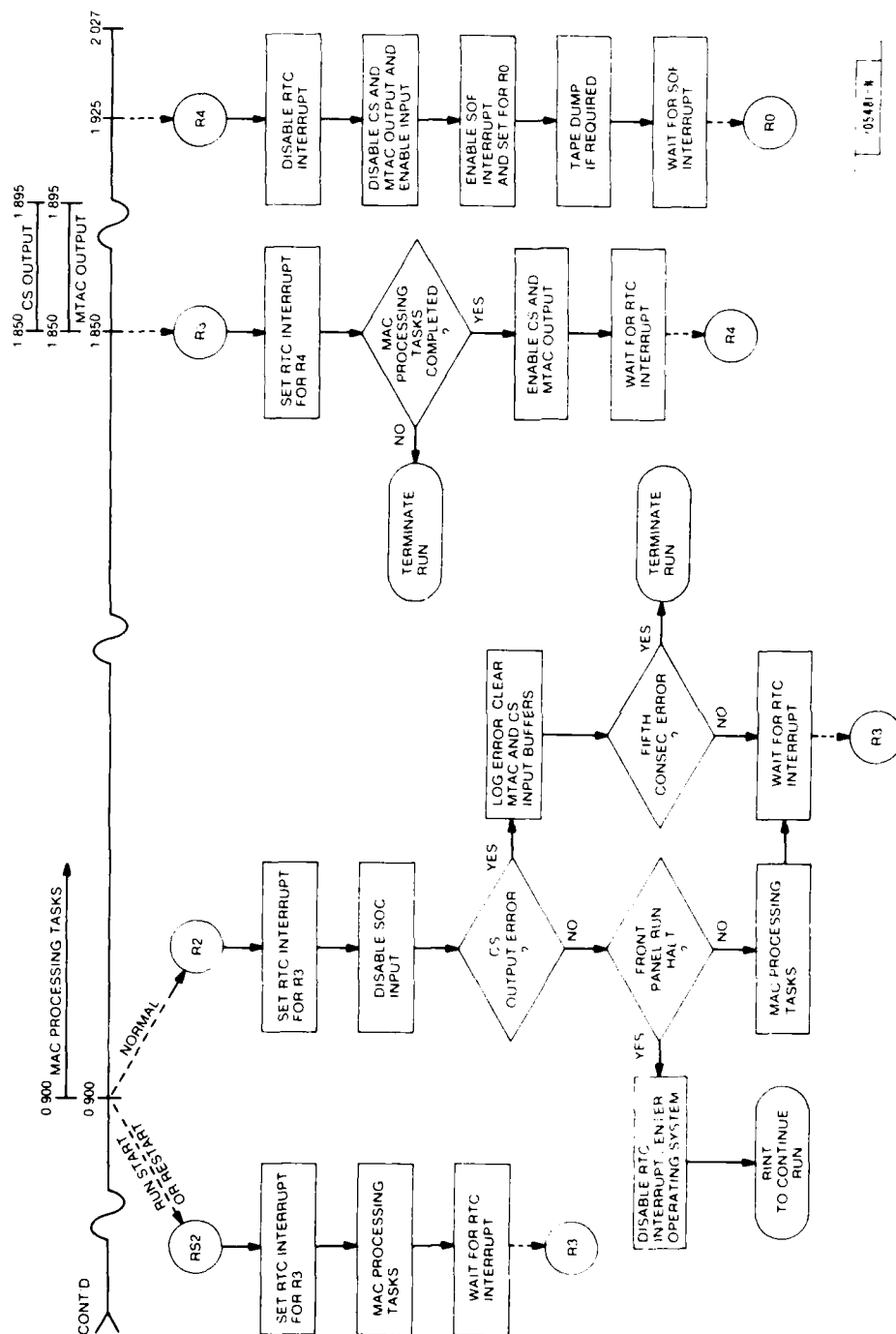


Fig. 2-2. Continued.

the time remaining in a frame after all I/O time has been accounted for. This eliminates the need for a processing-I/O overlap in the real-time structure, which would necessarily be far more complex than the current scheme. The MAC is now programmed to keep track of the maximum amount of single-frame processing time required during a run. Should the processing time provided for the MAC ever prove insufficient (which is not expected to occur), this fact will be recognized at event R3 and the run will be terminated.

A brief description will now be given of the sequence of real-time events. When the SOF interrupt occurs during a run, control is transferred to event R0. (Event RS0 occurs only at the first SOF interrupt after RINT, that is, when real-time mode is being entered.) At this time the BIC (the Varian DMA controller) is directed to transmit system status data to the System Operator's Console. The data are contained in a buffer set up during the MAC processing tasks in the previous frame. Input data from the Master TAC and the Call Simulator are expected immediately after SOF; hence the BICs have been prepared for input from these units during the previous frame (at R4). At R1 (0.700s after SOF), all input from the Master TAC and the Call Simulator should be complete, and the appropriate interfaces are disabled. A loop is now entered to sense completion of the transfer of data to the System Operator's Console. When completion is sensed, the BIC is directed to accept (system operator command) input from the console.

When the MAC was in the demand assignment performance testing phase, requiring long (overnight) runs, a problem developed in the MAC / Call Simulator serial interface that intermittently prevented the simulator from properly receiving MAC output. Replacement of the I/O controller cards used for the interface failed to correct the problem. Since the exact nature of the problem remained unclear, and further MAC performance testing was necessary, an "automatic repeat request" (ARQ) scheme was developed. This scheme is implemented at event R2 (SOF + 0.900 s) immediately after the System Operator's Console input is disabled. If the Call Simulator had not correctly received the previous frame's MAC (control burst) output, a special code is sent in the following simulator-to-MAC data stream. If this code is detected at R2, the MAC logs the occurrence of the output error and clears the Master TAC and Call Simulator input buffers of the data just received. The run is terminated with an error message on the fifth consecutive error; otherwise, MAC processing is suspended for a frame, allowing the previous frame's output to be retransmitted to the Call Simulator upon the occurrence of event R3.

If no indication of a Call Simulator output error is found at R2, a Varian front-panel sense switch is tested for the occurrence of a run halt. If the run is to be halted, the RTC interrupt is disabled and the Debug operating system environment is entered, allowing inspection of the MAC's data base and I/O buffers. When a run has been halted, the MAC's input buffers contain the data, just received from the three other Central Control Facility units, that

will be processed upon resumption of real-time operations (via RINT). The output buffers contain the data just sent to the units. All TACS units reflect the results (circuit assignments, preemptions, satellite ranges updates, etc.) of the last frame of MAC processing before the run halt, i.e., the entire system is in a consistent state.

If no run halt occurs at R2, MAC processing begins. Note again that no I/O occurs during this time. The data in the input buffers are merged and processed, the results are reflected in the output buffers and the MAC's data base, and the input buffers are cleared. MAC processing will be discussed in greater detail in sections to follow.

At SOF + 1.850 s event R3 occurs. If the MAC has not completed its processing tasks, the run is terminated with an error message. Otherwise, the output of (control burst) data to the Call Simulator and Master TAC is enabled. At R4 (SOF + 1.925 s) the Call Simulator and Master TAC interfaces are enabled for the input expected at the next SOF interrupt. A tape dump will occur at this point if a front-panel sense switch is enabled and the MAC's frame count is at a multiple of 1000. The data sent to tape, accumulated during the MAC's processing tasks, may be used to study the time behavior of the MAC's performance.

Examination of the TACS TDMA frame structure and the real-time and processing interactions of the major TACS units indicates system delays in the establishment and termination of circuits. Appendix B contains an example of the timing aspects of such an assignment and termination for a circuit of three frames duration.

E. Processing Structure

1. Considerations Influencing Processing Structure Design

The processing tasks of the MAC were organized with two TACS frame structure limitations in mind: (1) the limited amount of control burst capacity per frame, and (2) the limited amount of time that could be allotted for processing per frame. (Note that the latter has not, in retrospect, proven to be a constraining factor; see Section II.D.) The implication of these limitations is that the processing tasks must be ordered such that those with highest priority have first access to the MAC's time and control burst resources.

MAC processing tasks that are necessary to maintain system stability and consistency are of highest priority and are performed first. These include such tasks as using the subscriber units' status reports to update the MAC data base (to more accurately reflect subscriber communications activity and link quality), putting all new user requests into a queue for later servicing, and issuing satellite range corrections to subscriber units. The next level of priority is given to the system operator's commands, all of which alter the system's communications configuration. Finally, the MAC services the user

requests that are waiting in queue, altering its data base as it does so. The MAC continues at this task until either all requests have been serviced, processing time for the frame has expired, or all of the control burst capacity for the frame has been used up.

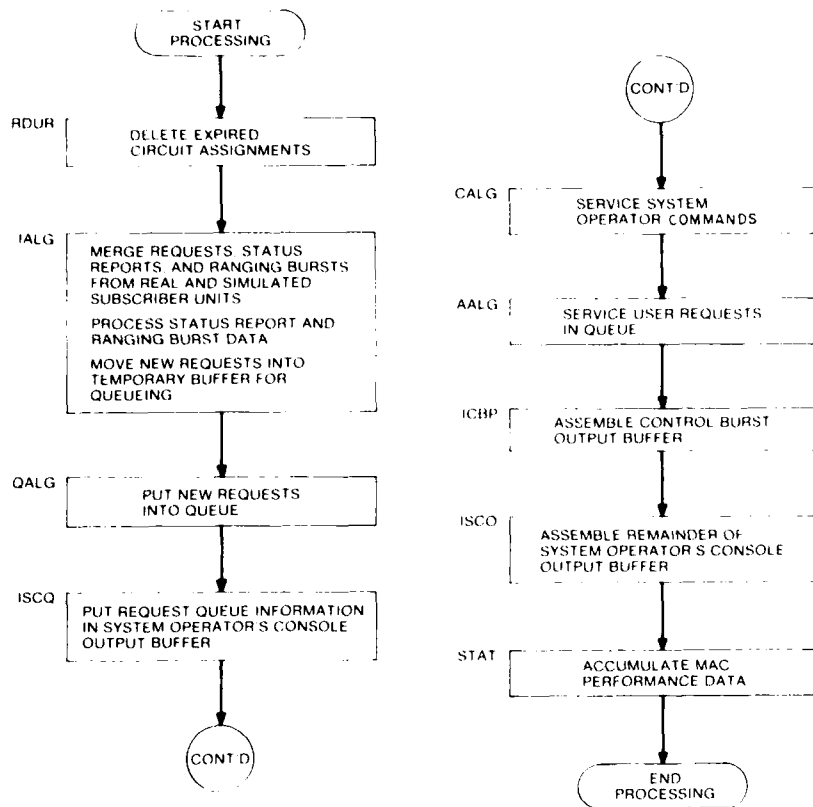
In some cases, the ordering of tasks has been due to their logical relationship, or due to the MAC's software structure, rather than priorities. For example, the routine that assembles the MAC's control burst output buffer is not run until all of the MAC's data base changes for the frame are complete, i.e., until the MAC has completed processing requests for a frame. Although this routine is of high priority (the MAC must output control burst data every frame) the requirement that the MAC reflect all of its data base changes in the control burst results in this routine being run as one of the final MAC processing tasks.

2. Description of Processing Structure

Figure 2-3 illustrates the sequence of the processing tasks that are performed by the MAC during every frame (at real-time event R2 or RS2; see Fig. 2-2). Each box represents a subroutine that is called by the MAC's main program. The labels next to each box correspond to the subroutine names. Figure 2-4 indicates, in tabular form, which segments of the MAC provide the primary input(s) and output(s) for each of the processing routines. These segments are divided into four categories: the MAC's input buffers, output buffers, temporary buffers, and data base. Note that the "MAC performance data", accumulated for the purpose of MAC performance evaluation, would not be part of an operational MAC. A detailed description of the contents of the input and output buffers can be found in Appendix C, "System Management Data Formats." Appendix E, "Multiple Access Controller Data Base," contains detail on the five major elements of the MAC's data base. The MAC has two "temporary" buffers: one holding newly arrived requests awaiting placement in the request queue, and one holding control burst messages awaiting placement in the next control burst.

The remainder of this section presents a brief functional description of each of the processing routines. The two most complex routines, the input and assignment algorithms (IALG and AALG), are described in greater (although not ultimate) detail in Sections II.F and II.G, respectively. An understanding of the functioning of the MAC's processing routines can be enhanced by reference to Appendices C and E.

The first task performed by the MAC each frame is the deletion from its data base of all circuit assignments whose time estimates have expired. (The time estimates may be supplied by end users when requesting circuits.) This is accomplished via RDUR. RDUR searches the MAC's assignment list for expiring assignments. All records of such assignments are removed from the MAC's data base, returning the previously assigned slots to the demand assignment pool.



105483-N

Fig. 2-3. MAC processing structure.

ROUTINE NAME	MAC INPUT BUFFERS			MAC OUTPUT BUFFERS		MAC TEMPORARY BUFFERS			MAC DATA BASE					MAC PERFORMANCE DATA
	MASTER TAC	CALL SIMULATOR	SYSTEM OPERATOR'S CONSOLE	CONTROL BURST	SYSTEM OPERATOR'S CONSOLE	REQUESTS	CONTROL BURST MESSAGES	REQUEST QUEUE	ASSIGNMENT LIST	TAC LIST	NET LIST	SATELLITE CAPACITY MAPS		
RDUR									10	0	0	0		
IALG	1	1				0	0			0				
QALG						1		0						
ISCQ					0			1						
CALG	1	1							0			0		
AALG							0	10	10	10	10	10		
ICBP				0			1							
ISCO					0				1					
STAT							1		1				0	

KEY 1 : PRIMARY INPUT
0 : PRIMARY OUTPUT

105484-B

Fig. 2-4. Primary inputs and outputs for MAC processing routines.

No control burst data are generated by RDUR, since all subscriber units are designed to independently terminate assignments when their specified durations have passed.

IALG is the input handling algorithm: It (1) uses the data input from the real TAC units and the Call Simulator to simulate any contention that may have occurred in the use of the random access request/report or ranging sub-slots provided in the TDMA frame, (2) does all of the required processing on the status report and ranging burst data that did not contend, and (3) moves all non-contending requests into a temporary buffer to await queuing.

Contention for the use of the ranging and request/report subslots must be simulated in the MAC, as the latter is the only link between the real and simulated TAC units. All ranging, request, and status report data sent to the MAC are tagged with an indication of the subslot used. IALG is designed to totally ignore the data associated with any subslot accessed by two or more TAC units.

The processing of ranging burst data involves altering the MAC data base (specifically the TAC list) to place the ranging subscriber unit in an "on" status. An initial range correction message is placed in the temporary buffer for the next control burst to inform the subscriber unit of its "on" status and of its uplink timing correction to account for satellite range. All satellite range corrections are based on the times of arrival, measured at the Master TAC, of ranging bursts or status reports received in the appropriate subslots provided in the TDMA frame.

IALG processing of a status report involves a number of steps. The MAC's (TAC list) record of the subscriber unit's link quality is updated according to estimates supplied in the status report, and a satellite range update message is assembled for the next control burst. Finally, IALG is designed to delete all circuit assignments from the MAC data base that are indicated in the status report as no longer being active. When this is done, a preemption message is placed in the temporary buffer for the next control burst to ensure that all parties to the terminated circuit cease communication.

QALG is the software package maintaining the MAC's queue of circuit requests. The position of requests in the queue determines the order in which they will be serviced by the MAC assignment algorithm (AALG, described below). At any point in time, the requests in queue are ordered primarily by their precedence (every request, and hence every circuit assignment, is tagged with one of five levels of precedence by the end user) and secondarily by their frames of arrival at the MAC. During conditions of heavy request rates, the limitation on each frame's control burst capacity can cause requests to be held over in queue for multiple frames. Every frame, QALG recognizes which requests still remain in queue and merges these with the newly arrived requests that have been placed in a temporary buffer by IALG.

ISCO performs the function of assembling the MAC request queue status information required by the System Operator's Console for display. This information is formatted and placed in the proper part of the System Operator's Console output buffer. The request queue itself is unaffected.

System operator commands are now serviced by the MAC via the command algorithm, CALG. All of these commands affect the MAC data base. No control burst data are generated for commands such as the addition of a satellite channel to the demand assignment pool; TACS subscriber units do not need to know of this event. Conversely, the deletion of a channel from the pool requires the preemption of any circuits assigned in the channel in question, and may, therefore, generate preemption messages for inclusion in the next control burst. The system operator commands are described in detail in Section IV.D.7.

The assignment algorithm (AALG) is the largest MAC software package. Every frame AALG services pending requests in the order it finds them in queue, attempting to make the required circuit assignments. This process continues until either all requests have been serviced or the control burst capacity for the frame has been exhausted. In attempting to service a call the MAC uses its data base to analyze such items as the capabilities of the transceivers of the parties to the call to access TDMA slots (given the type of net involved and each party's communications configuration and current activity), the quality of the satellite links to be used, and the availability of TDMA slots in the demand assignment pool. The precedence level of all on-going communications is considered in this process: The MAC has the ability to preempt lower precedence assignments in order to assign higher precedence calls. AALG minimizes preemption impact, i.e., preemptions occur only when absolutely necessary to establish a circuit, the circuit(s) being preempted always having the lowest possible precedence.

Requests are deleted from the request queue as they are serviced by AALG. If the request is successfully assigned a circuit, the new assignment is added to, and any circuits preempted are deleted from, the MAC data base. An assignment message and any required preemption messages are sent out in the next control burst. If an assignment could not be made, a "call progress" message (equivalent to a commercial busy signal), giving the reason for call blockage, is sent in the next control burst. Circuit assignments are never made unless all of the information required to inform the user community of the assignment (i.e., both assignment and preemption messages) can fit in the control burst. This maintains agreement between the MAC data base and actual TACS communications activity, clearly a prerequisite for an efficient and consistent assignment algorithm.

With the completion of the assignment algorithm, all MAC data base changes have been completed for the frame, and all of the control burst messages required to reflect these changes are being held in a temporary buffer. ICBP, ISCO, and

STAT, the final three software packages indicated in Figs. 2-3 and 2-4, are used to indicate the MAC's new status to the TAC units (real and simulated), the System Operator's Console, and the MAC performance analyst, respectively. ICBP reformats the control burst messages held in the temporary buffer, and places them in the control burst output buffer for transmission to the Call Simulator and Master TAC. ISCO sets up all parts of the System Operator's Console output buffer other than that already set up by ISCQ. This includes a list of the current circuit assignments, and information on the number of real and simulated subscriber units that have just accessed each ranging and request/report subslot. STAT uses the MAC's assignment list and control burst message buffer to accumulate the MAC performance data that may be periodically dumped to tape at real-time event R4 (see description of MAC real-time structure). The performance data include such entries as counts of the number of frames each slot in the demand assignment pool has been assigned to communications nets, and the number of control burst messages that have been previously sent out, by message type. Such information is of great use in analyzing MAC demand assignment performance.

F. Input Algorithm

This section presents a more detailed description of the MAC's input algorithm (IALG) than that given previously. The description generally follows the IALG flowchart in Fig. 2-5. Familiarity with the formats of the ranging burst, status report, and request data sent to the MAC (described in Appendix C), as well as the formats of the major elements of the MAC's data base (described in Appendix E) is assumed.

As indicated in Fig. 2-5, the first part of the input algorithm simulates contention among the TAC units (both real and simulated) for the use of the ranging and report/request subslots provided in the TDMA frame. Each ranging burst, status report, and request block in the Master TAC and Call Simulator input buffers contains fields (e.g., time of arrival) used by IALG to calculate the subslot in which transmission was attempted. This information is used to accumulate counts of the number of TACs that attempted transmission in each subslot. The input locks that were sent in subslots accessed by multiple subscriber units are not processed any further by IALG; in a full-scale system the MAC would not acquire the data in such subslots.

When no successful transmission was made in a subslot (attempt count of "0" or multiple), IALG determines whether the subslot was dedicated, in the TDMA frame being processed, to an individual subscriber unit for status report transmission. If so, the subscriber's TAC list entry counting sequentially missed reports is incremented. IALG compares this count to the maximum allowable number of such misses, and, if the number has been exceeded, places the subscriber in "off" status in the TAC list. Such action is based upon the requirement that each subscriber's transmit timing always be within the error

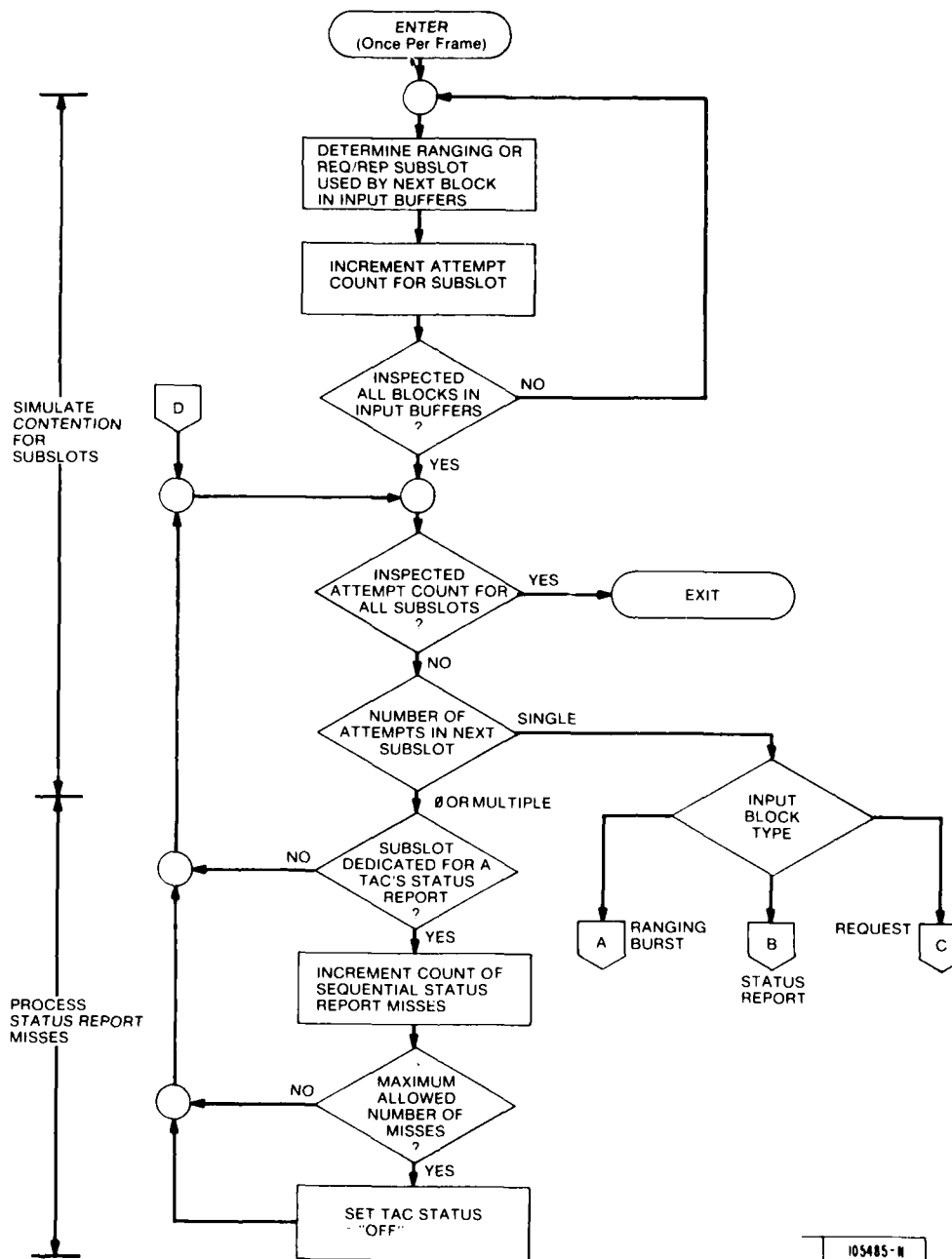


Fig. 2-5. Input algorithm (IALG) flowchart.

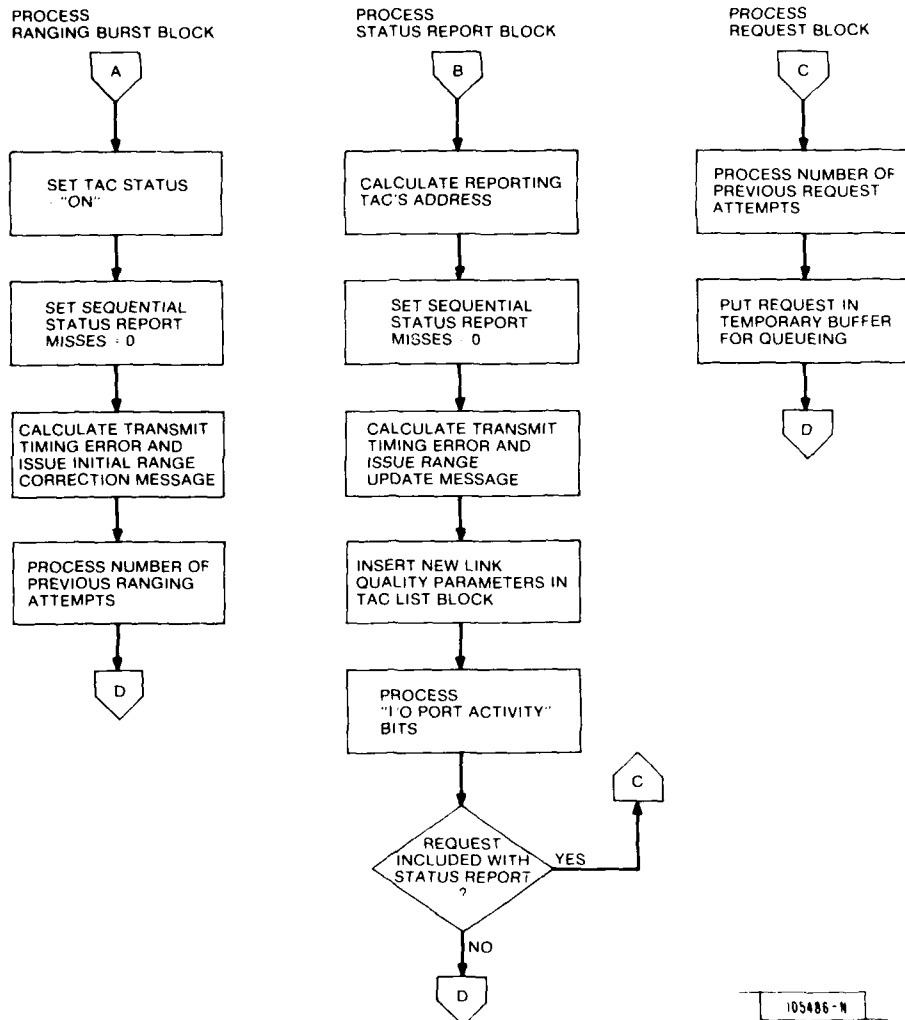


Fig. 2-5. Continued.

allowed by the TACS data slot guard intervals. IALG, using information derived from successfully transmitted status reports, generates range update messages that are placed in the control burst, allowing subscriber units to correct their transmit timing. If too many consecutive status reports have been missed, the error in a subscriber's transmit timing has potentially become great enough for it to interfere with other subscribers' transmission in adjacent slots. The subscriber unit recognizes that it must revert to "off" status (ceasing all on-going transmissions) when too long a period of time has gone by since its last range update. It must successfully transmit a ranging burst to the MAC in order to re-establish transmit timing and regain "on" status.

Input blocks transmitted in subslots with attempt counts of "1" did not contend with transmissions from other subscribers, and are processed by IALG according to their block type.

The first three steps of ranging burst processing shown in Fig. 2-5 should be obvious from the preceding paragraph's discussion. The "number of previous ranging attempts," included with the ranging burst block, indicates the number of times the subscriber had to re-transmit the ranging burst due to shared ranging subslots, and can be used by IALG to determine whether the subslots should be used in a shared or dedicated mode. (Note that, in a full-scale system, the MAC would not have the direct knowledge of subslot loading provided by the contention simulation. It must, therefore, determine the level of contention from the number of previous attempts.)

To process a status report block, IALG must first calculate the TAC address of the subscriber that had issued the report, the address being implicit in the subslot used for transmission. Information on the subscriber's link quality (uplink and downlink signal-to-noise ratio, and percent RFI) included in the report is used to update the appropriate TAC list block. Four status report bits ("I/O port activity" bits) inform the MAC of the current state of each of the subscriber's I/O ports. This information is used by IALG to delete circuit assignments that are no longer active from all parts of the MAC's data base.

IALG does little processing on request blocks. (The bulk of the request processing is done by the assignment algorithm.) The "number of previous request attempts" can be used to tailor the report/request subslot configuration to the current loading of these subslots. This could include the opening of secondary or tertiary sets of report/request subslots. IALG then reformats the request block, appending the frame of the request's arrival, and puts it in a temporary buffer to await placement in the request queue.

The Call Simulator provides the MAC with requests and status reports from simulated "remote" subscribers and from a simulated central control site. The latter would consist of nine subscriber units (and associated transceivers; one for each FLEETSAT channel) that would be hard-wired to the MAC. IALG simu-

KEY : C B : CONTROL BURST

[105487 N]

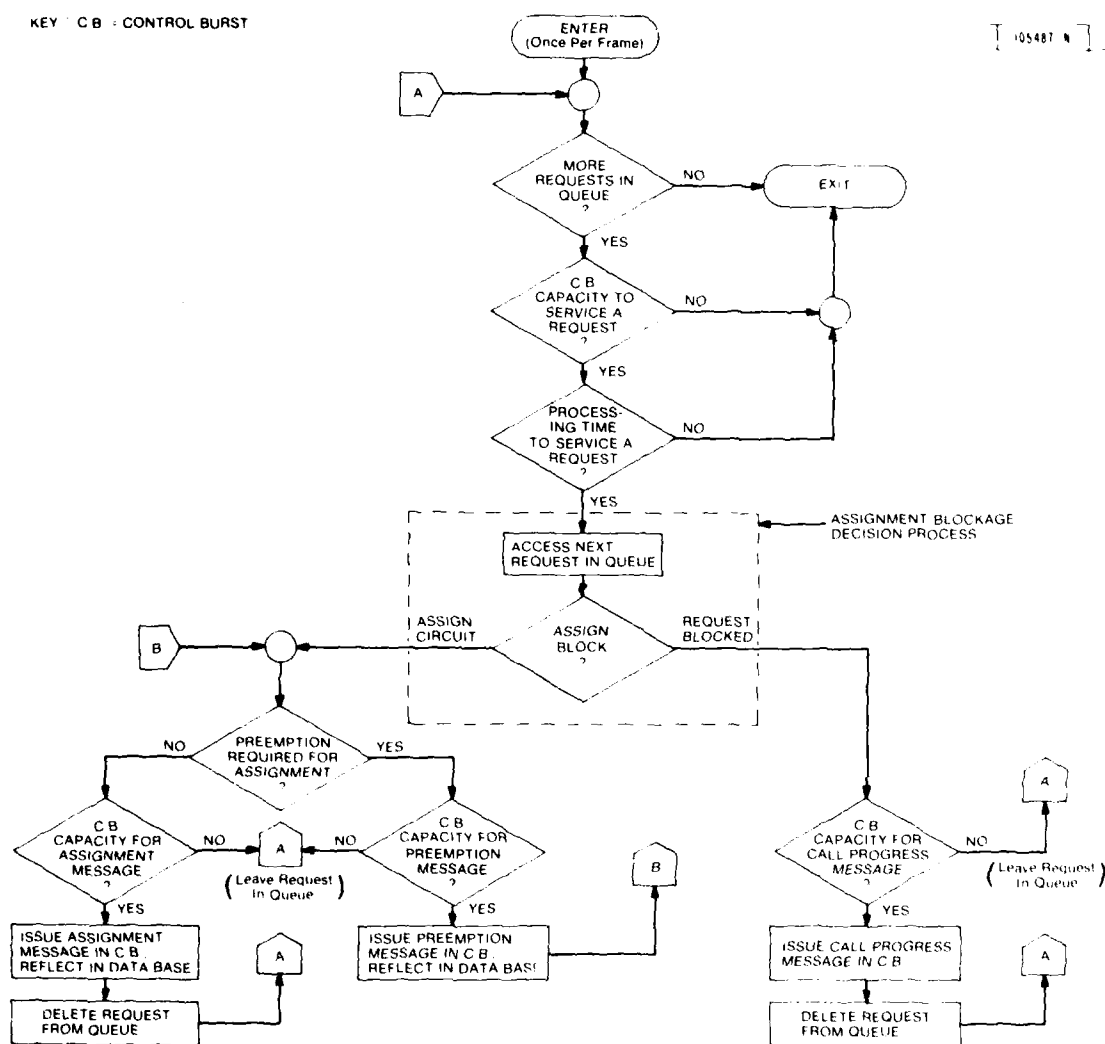


Fig. 2-6. Assignment algorithm (AALG) flowchart (emphasizing request queue and control burst management).

lates this direct link by recognizing requests and reports from the central site and bypassing the subslot contention simulation for such input blocks. As the TACs at the central site would provide the TDMA time base for the system, IALG does not issue them any satellite range corrections. All ranging features are also suppressed for the Master TAC, as it provides the TDMA time base for the two real TACs in the current testing configuration.

G. Assignment Algorithm

This section contains a more detailed description of the MAC's assignment algorithm (AALG) than given previously. The description is presented in three sections: (1) Assignment Algorithm Overview; (2) Assignment/Blockage Decision Process: All-Member Nets; and (3) Assignment/Blockage Decision Process: Point-to-Point Nets. Familiarity with the contents of Appendix C, "System Management Data Formats," and Appendix E, "Multiple Access Controller Data Base," is assumed.

1. Assignment Algorithm Overview

Figure 2-6 is a flowchart of the assignment algorithm, stressing AALG's management of its request queue input and control burst message output. Every frame AALG works its way through the queue of pending requests, attempting to assign the requests satellite capacity. As shown by the first part of the AALG flowchart, this process continues until all requests have been serviced, the control burst capacity for the frame has been exhausted, or processing time has expired.

Once AALG has determined the availability of the time and control burst capacity required to process the next request in queue, the "assignment/blockage decision process" is entered. Although shown as a single functional block in Fig. 2-6, this process is the most complex part of AALG. It contains two distinct branches: one for requests from all-member nets and one for requests from point-to-point nets. The two sections to follow present the detailed structure of these branches.

The assignment/blockage decision process draws upon all elements of the MAC data base. Its principal output is a binary decision: Assignment of system capacity to the request is presently possible or impossible. The decision is based upon the current status of all TACS elements (transceivers, I/O ports, satellite links, TDMA slots, etc.) along with the parameters of the request.

When circuit assignment is deemed impossible, an indication is given as to the reason for request blockage. The source of blockage is coded in a call progress message. As indicated in Fig. 2-6, a determination must next be made as to whether there is adequate control burst capacity for the call progress message. This test is required due to the message pooling arrangement used in the "elastic" area of the control burst, which contains limits on both the total amount of pooled control burst capacity (bytes) available per frame and the num-

ber of each kind of control burst message that may be sent per frame. The test at the beginning of AALG determines whether any control burst capacity remains; the capacity required to handle the (call progress, assignment, or preemption) messages reflecting the results of the assignment/blockage decision process for a particular request is not guaranteed.

If the call progress message can be placed in the control burst, the request is considered to have received final action and is deleted from queue. The source of request blockage coded in the message is ultimately displayed on the calling party's TIC. If no control burst capacity is available, the request is left in queue and will be serviced anew by AALG in a subsequent frame.

For a "successful" request the assignment/blockage decision process generates the assignment parameters (channel, slot, burst rate, code rate, etc.) used to create an assignment message. Indication is also given of any circuits that must be preempted in order to establish the assignment. AALG must first determine if there is adequate control burst capacity for each required preemption message. If the capacity exists, the message is placed in the control burst and the circuit being preempted is deleted from the MAC's data base. When there is insufficient capacity for a required preemption message, or for the assignment message itself, the request being serviced is left in queue to be processed by AALG in a following frame. An assignment will only be made if all required preemption messages and the assignment message fit in the control burst. In such a case the assignment is reflected in the MAC's data base (assignment list, satellite capacity maps, net list, TAC list) and the request is deleted from queue, having received final action.

Before presenting the details of the assignment/blockage decision process, a description will be given of the sources of both preemptions and call blockage (call progress messages) in the assignment algorithm.

Although all preemption messages are expressed in the same manner (by the channel and first slot occupied by the circuit being preempted), preemptions occur for three general reasons. "Circuit" preemptions arise from lack of the satellite capacity (idle TDMA slots in one of the channels being managed by TACS) required for circuit assignment. A lower precedence assignment may be preempted to free up the required capacity. "I/O port" preemptions are only generated in servicing requests from point-to-point nets. It is possible, in such cases, that the called party's I/O port is already engaged in communications activity with a third net member. This ongoing circuit may be preempted if its precedence is lower than that of the new request. "Transceiver" preemptions occur when one of the parties to a call is transceiver-limited; that is, when that party must terminate an ongoing assignment in order to free up the transceiver capacity required for the new (higher precedence) assignment. All-member net circuits are never transceiver preempted, as all or part of each net member's transceiver capacity is permanently dedicated to the net.

Figure 2-7 lists the nine call progress messages that have been established, along with an indication of whether they correspond to blockage encountered by requests from all-member or point-to-point nets. A "satellite busy" message results from a failure to find, in any channel and even allowing preemptions, the TDMA slots required to assign a circuit to an all-member net. "Terminal busy" indicates the same blockage for point-to-point nets, as well as any I/O port and/or transceiver blockage. (Note that the software structure of the point-to-point net request branch of AALG cannot distinguish between satellite and transceiver blockage.) "Terminal off" and "terminal covert" indicate that the called party's TAC is in "off" or "receive-only" status, respectively. An "insufficient link" message is issued whenever the worst satellite link of the parties to the request is bad enough that the TACS baseband error rate specification cannot be met, even at the lowest available burst and code rates. This message may also be issued when a TAC's link is not adequate to joint an ongoing assignment (via a conference or late entry request). An "invalid request" message is generated when there is an inconsistency in a request as received by the MAC. For example, a call to a TAC not in the calling TAC's (point-to-point) net would receive such a message. A "net active" message is issued when a party to an ongoing all-member net circuit makes an unsuccessful "override" request. For example, a particular net member may be one of the receivers of an ongoing all-member broadcast call. In order to transmit a message, this end user must generate a request that, if successful, will preempt the current assignment and establish a new (broadcast or two-way) circuit. If the override request's precedence is not high enough, a "net active" message will be issued to the end user. "Net inactive" messages are issued in response to late entry requests from members of inactive nets or in response to conference call requests that had been generated by parties to assignments that are no longer active. "In queue" messages are the only call progress messages that do not represent final action on a request. They are issued in certain cases where an assignment mes-

105488-N

CALL PROGRESS MESSAGE	NET TYPE	
	ALL-MEMBER	POINT-TO-POINT
SATELLITE BUSY	X	
TERMINAL BUSY		X
TERMINAL OFF		X
TERMINAL COVERT		X
INSUFFICIENT LINK	X	X
INVALID REQUEST	X	X
NET ACTIVE	X	
NET INACTIVE	X	X
IN QUEUE	X	X

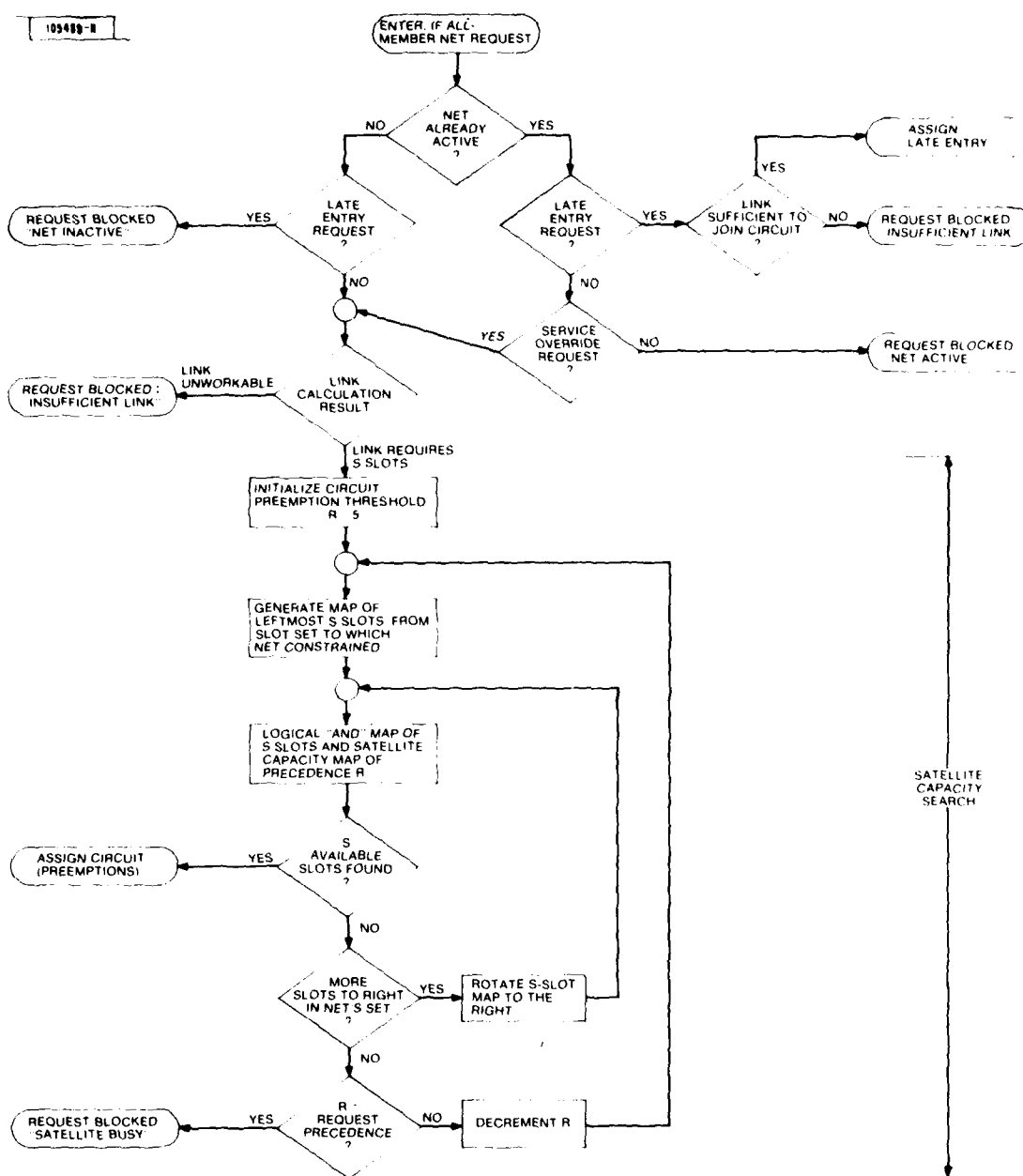


Fig. 2-8. Flowchart for assignment/blockage decision process for all-member net request.

sage and any accompanying preemption messages could not all fit in the current control burst. In queue messages inform the calling TAC of the MAC's reception of its request, preventing unnecessary cluttering of the shared request/report subslots with retransmissions of the request.

2. Assignment/Blockage Decision Process: All-Member Nets

Figure 2-8 is a flowchart of AALG's assignment/blockage decision process for requests from all-member nets. The terminators (ovals) on the left and right sides of the flowchart indicate the possible outcomes of the process: request blockage (indicated by call progress messages) or a decision to assign a circuit (along with any preemptions that may be required).

Three outcomes are possible for requests for late entry into a net. If the net is currently idle, the response is simply a net inactive message. If the net is currently assigned a circuit, the satellite link quality of the net member attempting a late entry must be examined. If the link quality is such that it can meet the TACS baseband error rate specification ($P(\text{bit error}) \leq 10^{-5}$ for record traffic and $\leq 10^{-3}$ for voice traffic) while working at the net's current burst and code rates, the late-entering TAC is issued an assignment message. As the circuit has been assigned previously, no preemptions are necessary. If the net member's link quality is not adequate to join the circuit, an insufficient link message is generated.

The MAC interprets a request for a new circuit received from a member of an active net as an override request. Two conditions must be met before AALG will further service an override request. First, the net must be currently operating a broadcast circuit, with the override requesting net member being one of the receiving parties. (Overrides are used by net members to gain the ability to transmit.) Second, as successful overrides preempt the net's ongoing circuit, the override request's precedence must be greater than the ongoing circuit's precedence. If either of these conditions is not met, a net active message is generated.

Before embarking on the search for satellite capacity with which to satisfy circuit requests from inactive nets (or override requests) a link calculation is performed. This involves determining the composite worst link (in terms of signal-to-noise ratio and RFI) of all of the net members, and using the link data for a table lookup to determine the burst and code rates that the net must employ to meet the system's baseband error rate specification. These burst and code rates, together with net's baseband data rate, determine the number, S , of contiguous slots required for the circuit. If the table lookup indicates that none of the available burst and code rates will provide adequate error protection, or if S is greater than the number of slots in the net's reserved slot set, an insufficient link message is generated.

The remainder of Fig. 2-8 flowcharts the search for satellite capacity, emphasizing AALG's preemption impact minimizing feature. The satellite capacity

search results in either a satellite busy message or a decision for circuit assignment (with possible preemptions). Note again that an assignment will only be made if the control burst has sufficient capacity remaining for an assignment message and any required preemption messages.

The algorithm for locating satellite capacity makes use of the satellite capacity maps (Fig. E-5). The search initially uses the precedence 5 (lowest precedence) map. If no available slots are found using this map, and if the request precedence allows (if it is ≥ 4) a new search is made using the precedence 4 map, thereby ignoring (virtually preempting) all assignments of precedence 5. This process continues until either the capacity to establish a new circuit is found, or the precedence level of the last map searched (the circuit preemption threshold, R , in Fig. 2-8) reaches the request precedence.

The search for satellite capacity is always confined to the set of slots reserved for the all-member net (at the members' transceivers). All sets of S contiguous slots within the reserved slot set are investigated for a vacancy in each of the nine satellite channels before the search over a given satellite capacity map is completed. The leftmost set of S contiguous slots is inspected for availability in each of the channels in order, followed by the next set of S slots to the right, etc. The search involves the generation of a "slot map" for each S -slot set. The slot map is similar in structure to the words (representing channels) of the satellite capacity map, since it is used to mask these words in the search for available slots.

"Leftmost" slots refers to the slots earliest in the TDMA frame. It is generally desirable to assign these slots preferentially as they are the slots which, due to TACS control and report/request subslot accessing requirements, are accessible to the fewest subscriber units. This saves the satellite capacity later in the frame for users with limited transceiver resources.

An example of the all-member net assignment algorithm processing is built around Fig. 2-9. The net member's transceiver capacity is reserved for this particular net during time slots three through five. This is shown in part (a) of the figure. Link quality for the worst-case net member is such that $S = 2$ slots are needed to guarantee reliable data reception by that terminal. The map of the first (i.e., "leftmost") slot pair to be searched within the reserved slot set is portrayed in part (b) of the figure. Searching progresses via masking the precedence S satellite capacity maps (shown in part (c) of the figure), one channel at a time from the top with the search map and looking for a match in positions three and four. In this example, no match is found during the first search attempt, so the search map is shifted one position to the right and slots four and five checked for availability. This time, a match is found in channel three. Since this is a first-fit algorithm, the searching process stops, and the net is assigned slots 4 and 5 of channel 3. Note that since we are in the precedence 5 satellite capacity map, these slots were totally idle; thus no preemption is required.

0	0	0	1	1	1	0	0	0
---	---	---	---	---	---	---	---	---

(a) NET'S RESERVED SLOT SET MAP

0	0	0	1	1	0	0	0	0
---	---	---	---	---	---	---	---	---

(b) FIRST "SLOT MAP" FOR SEARCH

	SLOTS								
	0	1	2	3	4	5	6	7	8
1	0	1	0	1	0	1	1	0	0
2	0	0	0	0	1	0	0	0	1
3	0	0	1	0	1	1	0	0	1
4	0	1	1	0	0	0	1	0	0
5	0	0	0	0	1	0	1	1	1
6	1	0	0	0	1	1	0	1	1
7	0	1	0	0	1	1	1	1	0
8	1	1	0	0	1	0	0	1	0
9	1	1	0	1	0	0	1	1	1

(c) PRECEDENCE 5 SATELLITE CAPACITY MAP

KEY: 1 = AVAILABLE

105490-N

Fig. 2-9. Example of all-member net assignment algorithm.

3. Assignment/Blockage Decision Process: Point-to-Point Nets

The branch of the MAC's assignment algorithm handling requests from point-to-point nets is generally similar to the all-member net branch since it addresses many of the same issues (i.e., circuit availability and link quality). The main difference is that the point-to-point algorithm includes the additional task of investigating the availability of terminal resources (i.e., I/O port and transceiver) for all parties to a request. In the event that some existing "local" traffic at one or more of the terminals blocks the request, that existing traffic may be preempted if it is of lesser importance.

Design of the point-to-point assignment algorithm was strongly driven by the desire that the impact of preemptions should be minimized. As a result, the algorithm is a highly iterative, tentative process. The sole preemption impact minimization criterion used is the precedence of old circuits being preempted

in order to establish the new circuit. The number of circuits to be preempted, for example, is immaterial; three precedence 4 circuits will be preempted even if preemption of a single precedence 3 circuit would accomplish the same effect.

The mechanism used to minimize preemption impact is the establishment of "preemption thresholds." Basically, a threshold is set to one of the five TACS call precedence levels, and all ongoing traffic of importance less than the current threshold setting is considered to have been deleted from the system, or "virtually" preempted. A separate, independent preemption threshold is maintained for each potential blockage location: the transceivers of the parties to the request and also the satellite. The thresholds are initialized to precedence level 5 (i.e., "lowest" precedence) for the first pass through the algorithm for a given request. Thus the MAC initially attempts to service that request without preemption. If at any stage a blockage is encountered, then the threshold for that location is relaxed, one precedence level at a time, until the blockage is relieved by virtual preemption or the threshold reaches the request's precedence. In the latter case, the blockage is unresolvable and the request is dismissed with a blockage (call progress) message. At times (e.g., in the case where the request's parties can work some common slot, but that slot is already occupied for all satellite channels) a decision must be made as to where to virtually preempt additional traffic in order to relieve a blockage: at the calling terminal, at the called terminal, or at the satellite. If any one of the preemption thresholds is set at a lower precedence level than all others, then it in effect directs this decision. Conversely, if two or more of the thresholds are "tied" at the lowest value, then the choice is made according to the following hierarchy: calling terminal, called terminal, satellite.

Note that all of the preemptions mentioned above are "virtual" in that no choice of particular circuits to be definitely preempted has been made. Each time a preemption threshold is relaxed, all traffic of the next higher precedence at the location is considered to have left the system. This is accomplished for a given terminal by simply ignoring all ongoing local calls of precedence below its threshold when determining which TDMA slots are available in the terminal in light of its ongoing traffic. In the case of satellite capacity, the MAC maintains two sets of satellite capacity maps indicating occupancy of each TDMA slot for each frequency channel. One set indicates the true occupancy and the other set is used as a scratchpad copy, with which to evaluate the effects of virtual preemptions. Each of the map sets consists of five complete maps, structured as shown in Fig. E-5. The first map of a set reflects only precedence 1 traffic, the second map indicates traffic of precedences 1 and 2, and so forth. To investigate the effect of preempting all satellite traffic of an incrementally higher precedence, the MAC need only direct its attention to the next map.

As the point-to-point branch of the assignment algorithm processes a given

request, it checks for the availability of resources in the following general order: I/O port, transceiver, and satellite. At times, however, transceiver availability is reconsidered after a first-attempt search for satellite resources. I/O port availability is confirmed first (and for all parties to the request) since it is the simplest determination. It is a one-time binary test; either the I/O port is occupied with higher precedence traffic, or it is not. If an old call must terminate in order to free an I/O port for the new one, then both the old call's transceiver and satellite resources are considered to have been freed also. In that sense, the virtual I/O port preemption feeds forward and (positively) affects the subsequent searches for transceiver and satellite resources. Next, transceiver availability is determined for each party to the request, individually. All local terminal blockages must be resolved before a search for satellite capacity can commence. This process of determining which, if any, of the frame's TDMA slots are accessible to a particular terminal based on traffic it is already handling is exceedingly complex. Description of the method finally implemented is relegated to Appendix G, which also includes an example for the sake of clarity. Upon confirming that the transceiver for each party to the request has at least enough adjacent slots available to support the call (possibly as a result of local preemptions) the MAC must then check for existence of some adequate common set of slots accessible to all of the call's parties. If there is inadequate commonality, then the preemption thresholds are used to direct the process of reconsidering the transceiver availability of one of the parties if traffic of one level higher precedence were to be preempted. Note that, as in the case of virtual I/O port preemptions, any virtual transceiver preemptions feed forward and are considered to have freed the satellite circuits. Once a common slot set adequate to support the new call and requiring minimum local preemptions is determined, the MAC searches for a frequency channel in which those slots are available. Again, if that search is unsuccessful, the preemption thresholds direct a repeat search for capacity, but under the condition that a bit more existing traffic may need to be preempted. The end result of this process (for a successful request) is that the algorithm has identified the particular set of TDMA slots in a particular frequency channel which if assigned for the new call, will result in preemption of calls at the lowest possible precedence levels throughout the system (i.e., from the satellite, from the called terminal, and from the calling terminal).

Having picked the optimal slot set and channel for the new call, the MAC can then determine which particular old call(s) will actually be preempted. There is no latitude in choosing which calls to preempt in order to free I/O port or satellite resources, so these decisions are simple and quick. On the other hand, transceiver resources may be freed by any one of a number of different combinations of preemptions. Once again a rather complex algorithm is used in making the preemption decisions and they are made separately for each

party. Basically, the MAC starts by "reloading" the given terminal with the traffic known to fit without contention for transceiver resources. This includes the new call and all old calls of higher precedence than the final value of that terminal's preemption threshold. Then, one by one and in order of decreasing importance, the MAC allows old calls of precedence less than the preemption threshold to "reoccupy" the transceiver. After bringing each call back, the MAC tests for contention for transceiver resources; any contention discovered results in the preemption of the old call which caused it - the last call brought in.

The discussion to follow is more detailed than that which preceded, and follows very closely Fig. 2-10, the flowchart of the point-to-point net branch of the MAC's assignment/blockage decision process. This flowchart is structured like its analog for the all-member net branch, with final outcomes for a request depicted in the oval terminators towards the margins.

Consideration of a point-to-point net request begins with three simple tests on called party status. Clearly calls cannot be successfully placed to parties whose terminals are in "off" status. Calls requiring transmissions by a terminal in "receive-only" or "emissions controlled" mode are similarly not honored. Though not indicated on the flowchart, a broadcast call to a receive-only terminal is allowed. An I/O port is considered "available" if it is either (1) idle, or (2) busy with traffic of precedence lower than that of the request. In the latter case, the old call must be preempted, freeing not only the I/O port, but in addition both local transceiver and satellite resources. The effect of this required local preemption on satellite resources is accounted for by expunging the old call from the scratchpad copy of the satellite capacity maps.

If the request is to add another party to an ongoing assignment (conference call request), the MAC must first confirm that the original call is still on the air. Through an unwise action on the part of the calling party, the original request could have specified a time requirement. As soon as that time expired, the circuit would be automatically relinquished, independent of whether the calling party had succeeded in bringing all desired parties into the conference call and accomplishing his communications needs. In order to join an ongoing call, a new party must have a link at least as good as the composite worst case link among the original parties to the call. The original assignment parameters, including burst and code rates, and the location of the circuit (TDMA slot(s) and frequency channel) are not changed in order to accommodate a new, link-disadvantaged party. Finally, the new party's transceiver must be available, possibly through preemption of ongoing calls, in order for that party to join the conference call. Since the original assigned location, in terms of frequency and time slot(s), cannot change, the determination of transceiver availability does not require iteration in order to minimize preemption impact on local calls. The preemption threshold is set equal to

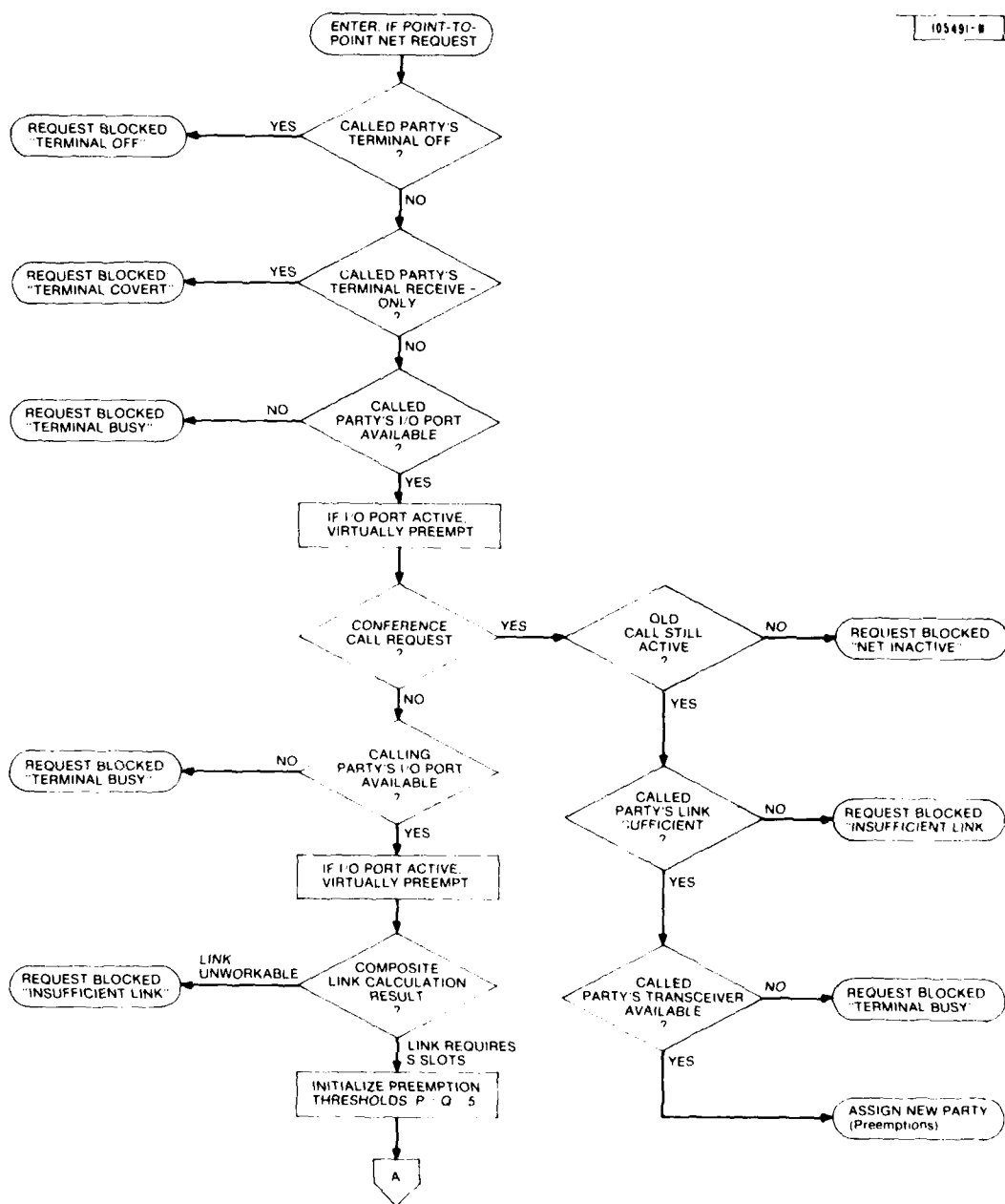
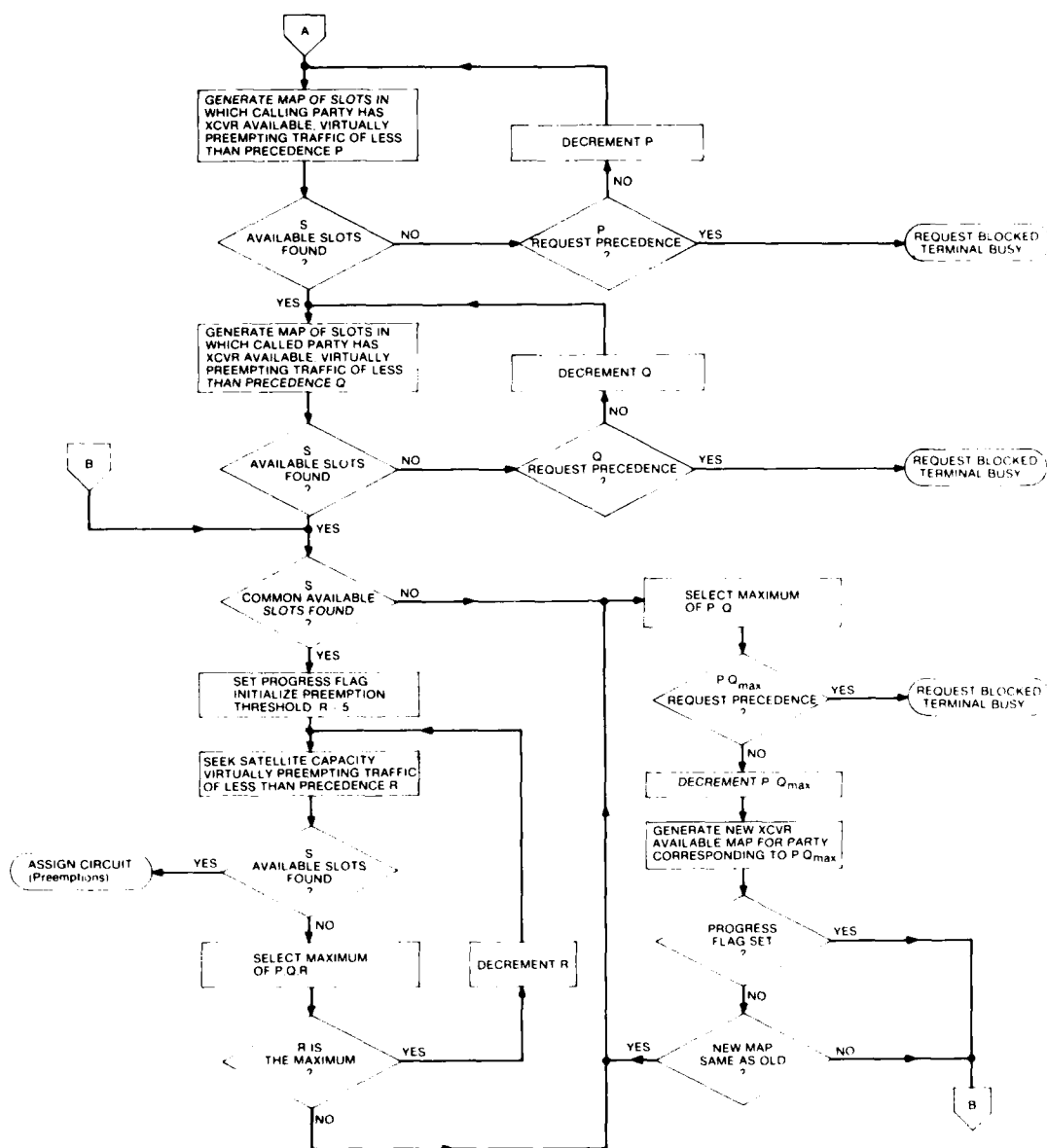


Fig. 2-10. Flowchart of assignment/blockage decision process for point-to-point net request.



105492-R

Fig. 2-10. Continued.

request precedence on the first (and single) pass through the transceiver availability subroutine.

If the request is for the establishment of a new circuit (i.e., is not a conference call request), then the MAC must first establish that the calling party's I/O port is still available. It may be assumed that the port was idle when the request was made, but it could have been occupied by an incoming call before MAC consideration of the request. Again, if an I/O port preemption is necessary, its effect on satellite resources is fed forward through a virtual preemption in the scratchpad satellite capacity maps. While investigating availability of I/O ports for all parties to the request, the MAC also inspects link quality, accumulating a record of the composite worst case values of S/N and RFI. In light of present values of these link parameters, the MAC attempts to choose the maximum burst and code rates such that the end user data will suffer bit error probability no worse than 10^{-3} (10^{-5} for record traffic). It is possible that even the lowest available rates cannot guarantee the required error performance, in which case the circuit request is denied. If the chosen burst and code rates are other than 32000 sps and 3/4, respectively, then the circuit requires a time interval consisting of more than one basic data slot (i.e., S slots on the flowchart).

Before commencing its search for transceiver resources for all parties to the request, the MAC initializes the preemption thresholds, with flowchart designations of P corresponding to calling party and Q corresponding to called party, such that first pass searches require no preemption. The two loops on the top of the second part of Fig. 2-10 depict the MAC's method of independently finding sets of at least S available (and contiguous) time slots for each party to the request, which slots, if assigned, will result in minimal local call preemptions. The method for computing which TDMA slots a terminal may access in light of existing traffic is detailed in Appendix G. It is in this section of software that the virtual I/O port preemption effects of feeding forward and freeing transceiver capacity are taken into account. Each time a terminal's preemption threshold is relaxed, the MAC generates a new map of available TDMA slots, corresponding to all traffic of one higher precedence level being dropped from that terminal. In some cases preempting all traffic of lower precedence than the request would still result in inadequate available transceiver capacity, so the request is blocked.

In order to participate in a call, all parties' terminals must be able to access the assigned time interval - requiring availability of at least S adjacent slots common to all terminals. The available slot maps previously generated, each one known to include a block of S available slots somewhere in the frame, are combined (by a logical "AND" process) and the result is tested for the required block of S common slots. In the event of failure to find a large enough time interval, the MAC attempts to rectify the situation by casting more ongoing traffic from one of the terminals. The terminal chosen

to suffer additional impact is the one whose preemption threshold indicates that, thus far in the assignment algorithm's process of progressive virtual preemptions, it has been subjected to less severe preemption. If the thresholds are equal, the calling party's terminal is picked. Should the preemption threshold corresponding to the terminal chosen during the present iteration equal the precedence of the request, then local blockages are irresolvable and the request must be denied. Otherwise, the appropriate preemption threshold is relaxed, and a new slot map is generated based on possible additional virtual preemptions. If the MAC has not yet found any set of S common slots (i.e., if the progress flag corresponding to this condition is not "set") and if the new and previous available slot sets for the terminal are identical, then the additional relaxation of the preemption threshold had not positive effect. Only when one of the terminals' maps changes is it fruitful for the MAC to retest for availability of a common S-slot time interval. If the MAC has, at some point while considering the current request, found a prospective common slot set and searched for available slots on the satellite, then it is always advisable to repeat the search for satellite resources. The reason: Even though any new local preemptions may have opened up no additional time slots for the terminal, they could have freed some slots on the satellite.

The mechanics of seeking satellite capacity (shown as a single block in the flowchart of Fig. 2-10) are similar to those described in greater detail in Fig. 2-8 for all-member nets. Subsets of S adjacent slots from the set of at least S available to all request parties are chosen, starting with the leftmost subset, and expressed in map form. This map is used as a mask on the satellite capacity map corresponding to virtual preemption of all satellite traffic of precedence less than R. The first match found is used to provide the circuit. It is possible, of course, that no available slots are found after an exhaustive search of the precedence R satellite capacity map. In that event, a difference between the point-to-point net branch and the all-member net branch of the assignment algorithm becomes apparent, and that difference relates to the system design objective of minimizing preemption impact. For all-member nets, the only way to continue considering the request is to allow virtual preemptions of satellite circuits at one level higher precedence (i.e., decrement R and search the next satellite capacity map). In the case of a point-to-point net, though, it is additionally possible to consider more transceiver preemption local to the parties to the request. As described previously, the MAC uses the preemption thresholds, in this case including R, to determine where virtual preemption impact has thus far been the least, and then investigates the effects of additional preemption there. The usual heirarchy applies in the case of equal preemption thresholds. If the decision is made to examine further transceiver preemptions, the MAC accesses the same section of software used when no common slots for the parties to the request had been found. This time through, the progress flag will have been set, and S common slots will be found, so the search for satellite resources definitely will occur. There are

three reasons why this new search might be more fruitful. First, additional preemption at one terminal may open more slots for that terminal, in turn making available more slots in the frame common to all request parties. Thus it is possible that more (conceivably idle) satellite slots would be candidates with which to fill the request. The possible existence of these idle slots is a reason for always starting the satellite resource search with the preemption threshold (R) set at 5. Second, even through additional preemption at one of the terminals may not open up more available slots, it may clear a call from the satellite. This is "free" capacity, requiring no additional preemption, and is one more reason why the precedence 5 satellite capacity maps are always searched first, even though the satellite preemption threshold (R) may have moved to higher levels during previous searches. Finally, even if allowing additional local preemption cleared no transceiver or satellite resources, the very fact that that the terminal's threshold was raised may allow raising of the satellite preemption threshold, thus opening up more slots to the request.

III. CALL SIMULATOR

The TACS Call Simulator generates requests for circuits ("calls") and status reports from a simulated large-scale TACS user community. The Simulator works from a "scenario," which is basically its own copies of the MAC's lists of net member addresses and of individual terminal configurations. Given a mean call rate per net in the scenario, the Call Simulator pseudorandomly generates calls, formats call parameters into request messages, and sends the requests to the MAC. The Simulator is quite realistic in the ways it formats and transmits requests, and in that it expects to get responses to its requests: otherwise it retransmits them. It is possible to drive the MAC with requests from both the Call Simulator and users at the real subscriber units.

The Call Simulator had three main uses: (1) During the MAC software development the Call Simulator was used to generate large amounts of test input, e.g., requests, which would exercise all assignment algorithm features, and hence aid in the discovery of programming errors. (2) Upon completion of the MAC, the Call Simulator used was to quantify numerous aspects of the MAC's performance in managing its pool of circuits. (3) There was great interest in demonstrating TACS under the combined demands of users at the two ("real") subscriber units built at Lincoln Laboratory and a more-or-less realistic communications scenario involving many other simulated users.

The earliest versions of the Call Simulator were programmed almost entirely in FORTRAN and were coresident with the MAC, in the Varian minicomputer. The request generation portion of the simulator was continually upgraded, throughout the duration of the TACS project, and the "Varian Call Simulator is still available. Its main advantage is that it requires no real-time structure (with incumbent waiting periods) in order to remain synchronized with the MAC, and hence allows itself and the MAC to run at speeds up to ten times as fast as could be achieved if the units had to conform to the 2-s TACS time frames. On the other hand, usefulness of the Varian simulator is somewhat limited since its interface to the MAC, its method of handling requests which must be retransmitted to the MAC, and its inability to generate status reports make it rather unlike a real user community.

Midway through the TACS development, the FORTRAN request generation segment of the Call Simulator was separated from the Varian computer and placed in a Digital Equipment Corporation PDP-11/03 minicomputer, eliminating any possibility of contention for processing-time resources between the MAC and Simulator. A real-time structure, appropriate request holding/retransmission software, and status report generation software, all written primarily in DEC's MACRO assembly language, was added, allowing almost perfectly realistic simulation of a community of users. The material to follow deals entirely with the more realistic "DEC Call Simulator," which has become a major element of the TACS Central Control Facility. Unique characteristics of the Varian Call Simulator are presented in Appendix H.

The Call Simulator software is even less optimized than the MAC's. FORTRAN, which is expensive both in required memory and run time, was heavily used because its high level nature speeded Simulator development. Even so, in the worst measured case, the Call Simulator minicomputer was only operating at 0.3 duty cycle; the other 70% of time available for processing was spent in a "wait loop." Total memory required for the Simulator is approximately 14K words; buffers and tables use 8500, the assembly language program uses 2500, and the 421-line FORTRAN program uses 3000 words.

As in the MAC and the System Operator's Console, the Call Simulator software exists in two forms, corresponding to the TACS five- and nine-slot TDMA frame structures. Again, the nine-slot version is the one most fully described, with unique characteristics of the five-slot version being listed in Appendix D.

The remainder of this description of the Call Simulator is divided into two subsections. The first is a functional description, which is strongly oriented around the inputs to and the expected outputs from the simulator minicomputer. The second, and much more specific, subsection details the way in which the simulator was implemented, relying heavily on block diagrams and flowcharts.

A. Functional Description

1. Overview

During each TDMA frame, the Master TAC unit passes all system control messages (i.e., circuit requests, status reports, and ranging bursts) received from the remote TAC unit to the MAC. A moderate amount of system control message processing must be performed by the Master TAC before it passes the messages along to the MAC. Each individual message is demodulated, deinterleaved, decoded, checked for parity errors, tagged with its time of arrival within the frame, and stored. Only when the Master TAC has accumulated the full complement of system control messages for the frame does it send them, in one burst during a short time window, to the MAC.

The function of the Call Simulator is to provide system control data bursts for the MAC, these bursts giving the appearance that there exists a large community of active, remotely located subscriber units communicating system control messages to the MAC via a master subscriber unit (or units). Since the main intent is to exercise and test TACS demand assignment rather than synchronization features, the Simulator generates requests and reports only, not ranging bursts.

2. Realism of Simulation

It was decided that the simulated system control message stream should be as similar as possible to the message stream generated by the real TACS subscriber units. Therefore, the individual simulated request and report formats were made identical to their real counterparts, and both real and simulated message streams are transferred to the MAC during the same time window. They enter the MAC through identical, but separate computer serial interface cards

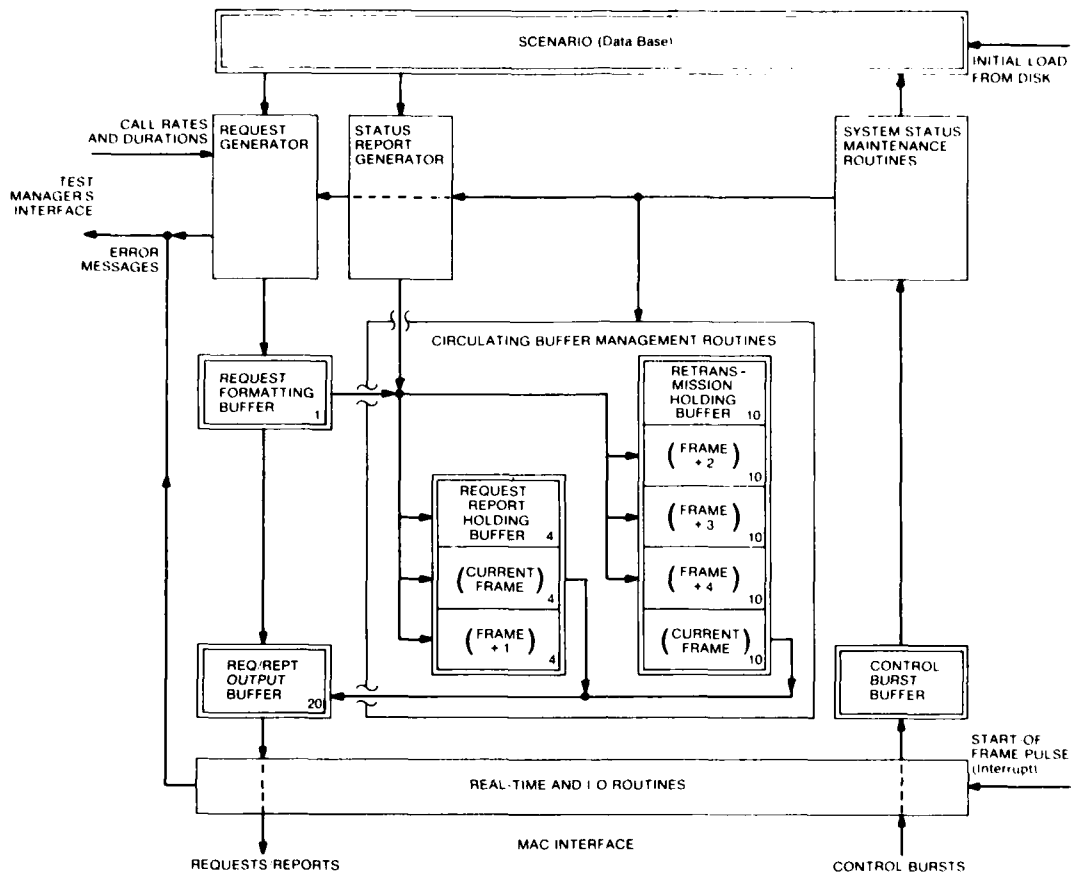
and are placed in separate buffers. After expiration of the time window, the MAC simply combines the two buffers of system control messages, and processes them all on an equal footing.

In addition to appearing quite like real subscriber units in terms of system control message formats and interface to the MAC, the Call Simulator is designed to conform to all TACS control Protocols. First when generating a circuit request, the Simulator cannot in a purely random manner pick values for the individual request elements (these elements include net address, call party address, call precedence, and several other items which are discussed in detail later) since to do so would result in inconsistent requests which would be rejected by the MAC. The values are "pseudorandomly" chosen - carefully made consistent with the MAC's view of system configuration and status. In order to maintain consistency, the Simulator actually keeps its own versions of several large tables which comprise the MAC's data base.

An example of consistency in configuration is evidenced by the method of picking calling party address. This request element is selected by a randomly driven pointer in the Simulator's look-alike version of the MAC's net list. Thus when the request is considered by the MAC, it can pass the test that the calling party is indeed a member of the net which that party is trying to activate. Similarly, an example of consistency in status is that already-busy users should generate no new requests for service: in queueing theoretical terminology, this is the "finite population" model. Having chosen a prospective calling party, the Call Simulator checks its up-to-date version of the MAC's TAC list, and if the party is already busy, the new request is suppressed.

The Call Simulator must also follow TACS control protocols when simulating transmission of requests to the MAC. Two features are worthy of special mention: anticipation of dedicated access request/report subslots and retransmission of requests which collide in random access request/report subslots.

As is the case with real requests, if a (simulated) terminal with need to transmit a (simulated) request will have dedicated use of a request/report subslot in either the current or one of the next two TDMA frames, it should use the dedicated access rather than using the shared subslots (and thereby unnecessarily adding to the contention for the shared accesses). For real terminals, this may require delaying request transmission for up to two frames. In the Call Simulator, this delaying function is accomplished by placing such a request in a "request/report holding buffer." There are three of these buffers, corresponding to the current and each of the next two frames. Each buffer is sized to hold four request/reports, four being the maximum number of dedicated accesses which TACS is designed to provide in one frame. If, for example, the simulated terminal is to have dedicated use of the third request/report subslot one frame hence, then the first new simulated request from end users at the terminal is held in the third position of the "frame + 1" request/



105473 - 0

Fig. 3-1. Call Simulator software/buffering structure.

report holding buffer. The holding buffers are used in a circulating fashion. At the beginning of each frame the simulator steps ahead a pointer which then designates what had previously been the "frame + 1" buffer as the "current frame" buffer. All request/reports in the current frame buffer are then automatically sent (contention-free) to the MAC.

In the case of real terminals, requests which cannot anticipate and use dedicated subslots are immediately (i.e., in the current frame) sent in the shared request/report subslots. There is of course a possibility that several terminals might choose to send requests in the same shared subslot, resulting in mutual RF interference, and in none of the requests being received by the MAC. The real terminals save a copy of any request transmitted in a shared access subslot. If they receive no MAC response to a request within two frames, they presume interference occurred, randomly chose one from the set of shared request subslots in the next three frames, and retransmit the request in that subslot. The Call Simulator accomplishes this function by placing a second copy of any request being transmitted in the current frame's shared request subslots in a "retransmission holding buffer". There are five of these buffers, corresponding to the current and the next four frames. Each retransmit holding buffer has capacity of ten requests. Usage of the buffers circulates in a manner similar to that described for the request/report holding buffers. To provide sufficient waiting time (i.e., a minimum of two frames) for response from the MAC, any requests generated in the current frame are placed, randomly, in one of the buffers of the three element set "frame + 2" through "frame + 4." Each request responded to by the MAC is deleted from the retransmit holding buffers. In the event of no MAC response, any request held eventually circulates into the "current frame" buffer, is once again sent to the MAC in a shared subslot, and is once again held for retransmission in case of further interference. It is worth noting at this point that the MAC determines, based on request arrival time in the frame, which of the simulated and real requests contended for the same subslots, and simply ignores all contending requests. Other than merging real and simulated system control data streams, this is the only extra burden placed on the operational MAC software by the requirement that the MAC must work with the Call Simulator.

3. Input Requirements

The complete software and I/O structure of the Call Simulator is depicted in Fig. 3-1.

a. Scenario

Before commencing a test run, the Call Simulator must be provided with information about the "scenario" to be used. This information consists of a pair of lists from the MAC's data base: the TAC list and the net list. The scenario information is required, because the MAC and Simulator must agree upon the configurations of all subscriber units and nets. The calling party address, one of

the elements of a request, must correspond, for example, to an address in the MAC's net list; otherwise the MAC will simply reject the request as invalid. The scenario is generated during an off-line (i.e., non-real-time) interactive episode between the system operator and a Varian program called the "Scenario Generator," more fully described in Section V of this report. At the end of a scenario generation episode, the tables constituting the scenario are dumped to a magnetic tape for later use with the MAC, and to floppy disk for use by the Call Simulator.

b. Statistics

Another set of test run initial conditions is the mean per-net call rate and call duration arrays. As part of its initialization phase, the Call Simulator prompts the test manager for these parameters via its console terminal keyboard. Individual nets may be "shut off" simply by specifying a call rate of zero. Once the test run commences, all requests and some parameters of the requests are generated on a pseudorandom basis. Intervals between calls, for each net, are Poisson distributed, with mean value as specified by the test manager. Similarly for individual calls, message durations are exponentially distributed. These distributions were chosen primarily because they correspond to those used in the development of most analytical queueing models to which TACS performance may be compared. The algorithms used to generate data which conform to these distributions are as described in Knuth.⁷

c. Control Bursts

Throughout the test run, the Call Simulator accepts the MAC's control burst data for each frame, in real time, and uses the information contained therein to maintain an updated record of system status. In terms of large data base blocks, status includes the TAC and net lists, and in addition a look-alike copy of the MAC's assignment list. Also extracted from the first few bytes of the control burst (see Figs. C-1 and C-2 for control burst format) are a number of single-word status entries, including frame-of-day and the current shared/dedicated mix of request/report subslots.

d. Start of Frame Pulse

A final input to the Call Simulator is the real-time start of frame pulse. This pulse is interfaced to the Simulator as an interrupt and is provided by the Fast Processor section of the Master TAC. The start of frame pulse is used to synchronize all of the Central Control Facility's units which have interrupt hardware to coordinate data transfers (i.e., the Master TAC, the MAC, and the Call Simulator).

4. Output Requirements

The Call Simulator actually transfers to the MAC a single buffer of output each TDMA frame. This buffer varies in size from zero to a maximum of 20 elements, each element being comprised of five 16-bit computer words. The buf-

fer's composition varies from frame to frame, but may include elements of four distinct classes: requests generated in the current frame and being transmitted for the first time in shared subslots, requests previously generated and being retransmitted in shared request/report subslots due to assumed earlier contention (in shared request/report subslots), requests generated in either the current or a previous frame but combined with status reports and being transmitted in dedicated request/report subslots, and finally simple status reports utilizing dedicated subslots. The following paragraphs describe the classes of output in further detail. Exact formats of requests and status reports are presented in Appendix C.

a. Requests

Each request which arrives at the MAC consists of a number of fields of related bits or "parameters," which are listed in Table 3.1 along with the range and nominal distribution of each parameter.

TABLE 3.1
REQUEST PARAMETERS

<u>Parameter</u>	<u>Range</u>	<u>Distribution</u>	<u>Flowchart Designation</u>
Net address	1 to 127	Fixed	
Calling party's TAC address	1 to 511	Uniform	CRAD
Called party's TAC address	1 to 511	Uniform	CEAD
Call precedence	1 to 5	Uniform (skewable)	PREC
Call duration	1 to 31	Exponential	MDUR
Late entry to net	0 or 1	Skewed (5%)	LECF
Conference call	0 or 1	Skewed (20%)	LECF
Call direction (Broadcast/two-way)	0 or 1	Skewed (mostly two-way)	DIRN
Number of previous request attempts	0 to 7	Fixed	

All request parameters other than those listed as being of fixed distribution are chosen, for each individual call, in a pseudorandom way (i.e., the result of values returned by a uniform random number generator) subject to certain constraints imposed by the requirement of realistic behavior.

In each frame, the order in which the Call Simulator generates requests corresponds to the order of the nets in its net list. That is, starting with the first net in that list, and for each net whose mean per-net call rate is non-zero, the simulator generates a number (possibly zero) of Poisson distributed events, each one of which corresponds to a new request. For those requests, the net address is naturally fixed. One of the first parameters chosen (and, incidentally, a parameter which affects the choices of a number of sub-

sequent parameters) is the TAC address of the calling party. This address is picked at random from the set of addresses of all members of the net. The range of this parameter is TACS system-limited to a maximum of 511, and may be further limited by the actual number of TACS in the scenario (63, 127, or 255). The TAC address of the called party is handled similarly, except that for all-member nets it is not used (set to zero) and for point-to-point nets it (naturally) cannot be the same as the calling party address. Call precedence is nominally distributed uniformly in the range one (highest) to five. Provision was included to skew the distribution towards higher precedences, but that feature was very seldom used. Certain special types of requests have restricted precedence ranges. Late-entry-to-net requests, for example, are forced to precedence level five, and requests to add parties to an on-going assignment (conference calls) are forced to conform to the initial call precedence. Call (or message) durations are exponentially distributed with per-net mean values as specified upon test run initialization. The value of zero is not allowed, since it would be interpreted by the MAC as a circuit need of indefinite duration, and a resulting assignment would never clear itself from the system. The raw call duration number corresponds to time frames for record traffic nets and to minutes for voice nets. The late-entry-to-net parameter is applicable only to all-member nets, with the same bit position in the request format being alternatively used as a conference call flag to add parties to an on-going assignment in a point-to-point net. Five percent of all-member net requests are of the late-entry type. In addition to precedence, two other late-entry request parameters are forced; call duration and call direction both equal to zero. If the chosen calling party in a point-to-point net is already busy, the Call Simulator will with probability 0.8 simply give up on generating the request, this action corresponding to the "finite population" model where busy users generate no new demands. The other 20 percent of the time, a conference call request is generated, and again a number of parameters are forced: calling TAC, precedence and call direction to their initial-call values, duration to zero, and called TAC to a non-participant in the initial call. The call direction parameter is another which is handled differently depending on the net type. Eighty percent of all-member net calls and sixty percent of point-to-point net calls are two-way, and the remainder are broadcast calls. The final request parameter from Table 3-1, a count of the number of previous unsuccessful attempts by the TAC to send the request to the MAC, is initialized to the value zero. A three-bit parameter, it saturates after seven retransmissions, but the range has proved entirely adequate. In a TACS environment configured with a sufficient number of shared request/report subslots, it should be rare that a request need be sent more than twice. Having generated the above parameters (net address through attempt count in Table 3-1), the Simulator has fully specified the request itself. The request parameters are then packed into appropriate fields of the five 16-bit computer words which constitute the request formatting buffer.

A real TACS subscriber unit prompts the calling user for most of the fore-mentioned nine request parameters. When the prompting sequence is complete, the terminal packs the parameters into a 40-bit stream, computes and appends an 8-bit parity-check (block) code, and transmits the request over the satellite links. The Master TAC, upon receiving the request burst, checks and strips off the parity bits, but also appends the request's time of arrival in the frame, frequency channel used, and data characterizing the remote terminal's uplink quality. Only after adding the time of arrival and uplink quality information to the request does the Master TAC pass it to the MAC's input buffer.

The Call Simulator interfaces directly to the MAC, without the aid of a Master TAC. Hence the Simulator has been designed to assume the Master TAC functions by appearing to measure request burst arrival time and data quality.

For the Simulator, picking the request arrival time (and channel) really amounts to choosing in which of the request/report subslots to "send" the request. That subslot number is then used as a pointer in a table which specifies, for each subslot, arrival time and frequency channel. The range from which the subslot may be chosen for request transmission is highly dependent on the instantaneous configuration of the system. If the request is going through the dedicated report/request subslots, the particular subslot used is dictated to be the one dedicated to the calling TAC. That subslot must be in the range 1 through N, where N is 1, 2 or 4, and corresponds to the number of subslots per frame which the MAC is using in dedicated mode. If the request is employing the shared request/report subslots, the particular subslot used is randomly chosen from the set of subslots currently designated for shared usage. In the final implementation of the TAC system, this is the remainder of the request/report subslots. The total count of request/report subslots may be 6, 12, or 18, depending on whether the MAC is currently providing additional request capacity by opening secondary and tertiary channels to requests. Request capacity is provided in increments of six subslots, the number which will fit within the boundaries of one TACS 2400-bps data slot. (See Appendix A, "Frame Structure Design.")

Choosing the particular subslot in which to send the request is part of the larger task of determining how to dispose of a new request currently resident and partially formatted, in the request formatting buffer. First, all of the requests being held in the Simulator awaiting response by the MAC are compared to the new request. If the new request is a duplicate (i.e., a subsequent request from the same end user, generated before the MAC has responded to a pending request) then it is discarded, since the real TACS subscriber units are programmed to allow end users to have only one request at a time pending. Since casting out requests perturbs the temporal distribution from the desired Poisson case, a counter is incremented each time a request is so discarded. At the end of the simulation, the test manager may inspect this perturbation log, and make a value judgment as to whether duplication of requests has invalidated the test,

for his purposes. Second, provided the new request was not a duplicate, the Call Simulator must pick one of two paths by which to send the request to the MAC. These paths are depicted emanating from the request formatting buffer in Fig.3-1. If the calling party's subscriber unit expects its dedicated report/request subslot to come around in either the current or one of the next two time frames, then the request simply waits, is combined with the unit's status report and takes advantage of the dedicated access to the MAC. Otherwise, the request vies for a random-access subslot during the current frame and a copy is saved in a holding buffer for retransmission after a wait of two, three, or four frames (randomly chosen) if no response is received to the first attempt. In the event that the chosen retransmission holding buffer is full, the request cannot be held for retransmission. This again is a perturbation from desired Simulator performance, and is logged in a manner similar to the handling of duplicate requests. In the very special case of a request emanating from a user at the simulated central control site (the Simulator considers TACs number 1 through 9 to be at the central controller), the request is combined with a report if a report is scheduled for the current frame; otherwise the request is passed directly to the MAC, with no copy retained. Retransmission is never necessary since central control site requests are local to the MAC, need not be relayed by satellite, and hence are not subject to contention.

As previously mentioned, uplink quality is not technically a request parameter. For TACS requests from real subscriber units, it is measured by the Master TAC and appended to the request, is then 8-level quantized by the MAC and finally stored in the MAC's TAC list along with other information pertaining to the calling unit. For simulated subscriber units, the (static) quantized uplink quality record is taken from the Call Simulator's copy of the TAC list, is "unquantized" using a table, and is appended to the request. The MAC reverses this process, forcing both MAC and Call Simulator to agree on the simulated users' link qualities.

b. Status Reports

The second output requirement of the Call Simulator is to generate status reports, at appropriate times, from all subscriber units which are active (in "on" status) in the simulated scenario. The report may stand alone or, if some one of the users at the subscriber terminal has a pending request, may overlay with the request. Report information consists of a total of nine bits - five describing the quality of the terminal's downlink and four indicating the status (whether currently engaged in communications or not) of each of the four I/O ports. All of these bits for a given simulated TAC are static, and as shown in Fig. C-3, they occupy the same bit positions as the calling party's address for requests. The link quality information is specified when the scenario is created using the Scenario Generator and is loaded with the TAC list during Simulator initialization; the Simulator simply extracts the information from its TAC list and places it in the report. Thus, to the MAC, it appears that all subscriber's

links remain at the value initially estimated. The "I/O port activity" bits are always "set," indicating the continued use of all circuits. As mentioned previously, all simulated calls are provided with time requirement estimates, and they leave the system automatically when that time expires. In the case where a report and a request are to occupy the same subslot, the report information overwrites the calling party TAC address of the request. That information is expendable, since the MAC knows (i.e., can calculate the address of) which TAC has use of each dedicated request/report subslot. The Call Simulator appends the time of arrival and uplink quality fields to reports, in the same formats used for requests.

After a unit's status report is generated, it is placed in the position of the request/report holding buffer for the current frame which corresponds to the subslot dedicated to that subscriber. The holding buffers each have a capacity of four reports (or report/requests), equalling the maximum number of subslots which the MAC can dedicate. During a given frame, the current frame's buffer will most likely contain less than four reports, for two reasons. Sometimes the MAC is configured to dedicate only one or two of the six request/report subslots. Secondly, the dedicated subslots are given to subscriber units with adjacent (integer) addresses, and not all units are "on" and generating reports.

c. Combined Request/Report Message Stream

For each frame, the first elements placed in the 20-element-capacity request/report output buffer (Ref. Fig. 3-1) for transfer to the MAC are any requests scheduled for retransmission. The imbedded count of the number of previous request attempts was incremented when the requests went into the retransmit holding buffer, so the held requests are simply copied over. There is naturally a chance that the retransmitted requests still will not succeed in getting through to the MAC, so they should be once again held for retransmission in a randomly chosen frame in advance. If there is space in the chosen retransmit holding buffer, the request's attempt count is again incremented and it takes a position in that buffer. Conversely, if the chosen buffer is already full (i.e., holding ten requests), the request cannot be held. That request is lost from the retransmission scheme, once again a perturbation from desired Simulator behavior, and as described previously a record of this perturbation is logged for post-mortem simulation run inspection. The next elements placed into the output buffer are requests newly generated in the current frame and vying for shared request/report subslots. The Simulator already knows the number of reports for the current frame and has reserved space in the output buffer, but has not copied the report/requests to that buffer yet. New requests continue to occupy the output buffer until either the request generation segment of the Simulator finishes its job for all nets in the scenario or until the output buffer saturates. In the latter event, the generator must stop prematurely - one more perturbation. A counter is incremented once for each TDMA frame during

which this happens. The final elements to be transferred to the output buffer are any reports or combined report/requests.

d. Perturbation Log (Internal)

A third output of the Call Simulator is a log of perturbations encountered during the course of a simulation. Unlike requests and reports, this log is not output in real time but is stored in Call Simulator memory and may be inspected after the simulation run is halted. The perturbation events concerning call statistics were described in the preceding paragraphs, and number four in type:

1. Duplicate requests
2. Requests which could not be held for their first retransmission
3. Requests which could not be held for subsequent retransmissions
4. Frames during which the output buffer was too small to accommodate the number of new requests being generated

In addition to this statistical perturbation log, the Call Simulator does a moderate amount of checking for internal errors, especially related to its real-time software performance. In the event that an error is detected, the Simulator either corrects the error and logs the event (e.g., in the case of a missed control burst) or aborts the run. After initial software checkouts, run aborts became virtually nonexistent, occurring only during severe AC power service disruptions.

5. Limitations

Though the actions of the TACS Call Simulator, as implemented, emulate a user community in a quite realistic manner, there do exist certain limitations in the scope of activities simulated. Specifically, the Simulator does not deal with initial ranging bursts, active call termination, and the time-variability of some terminal characteristics. Also, the actions of the Call Simulator and of the MAC in dealing with simulated request and report bursts are not apportioned to the two units in the same manner as they would be if all users were real.

a. Ranging Function

It was decided not to require the Call Simulator to pseudorandomly "turn on" subscriber units and, as part of the startup process, to send initial ranging bursts to the MAC. Rather, all subscriber units which are configured to be members of any net are initialized in an "on" state in the TAC list by the Scenario Generator. Thus these units are indicated as "on" when the data base is initially loaded into the Simulator, and the Simulator generates reports from the units throughout the simulation run. The reason for not simulating the initial ranging process is that performance behavior of the ranging subslots can be treated quite easily analytically.

b. Circuit Relinquishment

Real calls assigned circuits may leave TACS either upon expiration of the time requirement if specified in the request or upon receipt by the MAC of a report containing a "hangup" signal (in the "I/O port activity" field). All simulated requests include a (non-zero) time requirement, and thus all simulated calls terminate by the former method. The choice of specifying call duration at the time of request generation was made for ease of implementation. The hangup flags in the subscriber units' reports are never used by the Simulator (i.e., they are static) since it is undesirable to terminate a call prematurely, hence perturbing the exponential distribution of call durations.

c. Dynamics

An actual terminal's link qualities and uplink timing would vary as a function of time. Since TACS was a hardware development project, the effects of these variations could be studied using the real subscriber units, so for simplicity's sake the Call Simulator generated static link reports and burst arrival times. The Scenario Generator requires that estimates of uplink and downlink signal-to-noise ratios and pulse-blanker duty factors (percent RFI) be made for all terminals active in nets. This information is coded into a block of eight bits and placed in the initial-condition TAC list. Each time that a simulated terminal must send a link report, the Simulator simply goes to the TAC list and repeats the initial link estimate. The Simulator contains a table of six burst arrival times (relative to the start of frame), corresponding to the six request/report subslots. Independent of terminal or time of day, all bursts destined to use a given subslot are assigned the same arrival time, as specified by the table.

Continuing with the issue of burst arrival time, with real subscriber units, that time is measured by the Master TAC at the central control site and is appended to the request or report (or ranging) data before transfer to the MAC. Since the Call Simulator and MAC interface directly, that function had to be assumed by one of those units. The Call Simulator was the logical choice since appending time of arrival there allows identical formats for all requests and reports, real and simulated, arriving at the MAC. An extra burden which was placed on the MAC is the handling of contention for capacity in the shared request/report subslots. In a real full-scale TACS environment, all requests sent in the same subslot by various terminals would interfere at RF, and in all likelihood none would be successfully received. For the TACS demonstration system, the MAC uses request burst arrival times to calculate which subslot was used by each request (both real and simulated), and in the case where multiple requests choose the same subslot, casts out (i.e., does not place in queue) all of them. (For more detail on this process, see the description of the MAC "input algorithm" in Section II.F.)

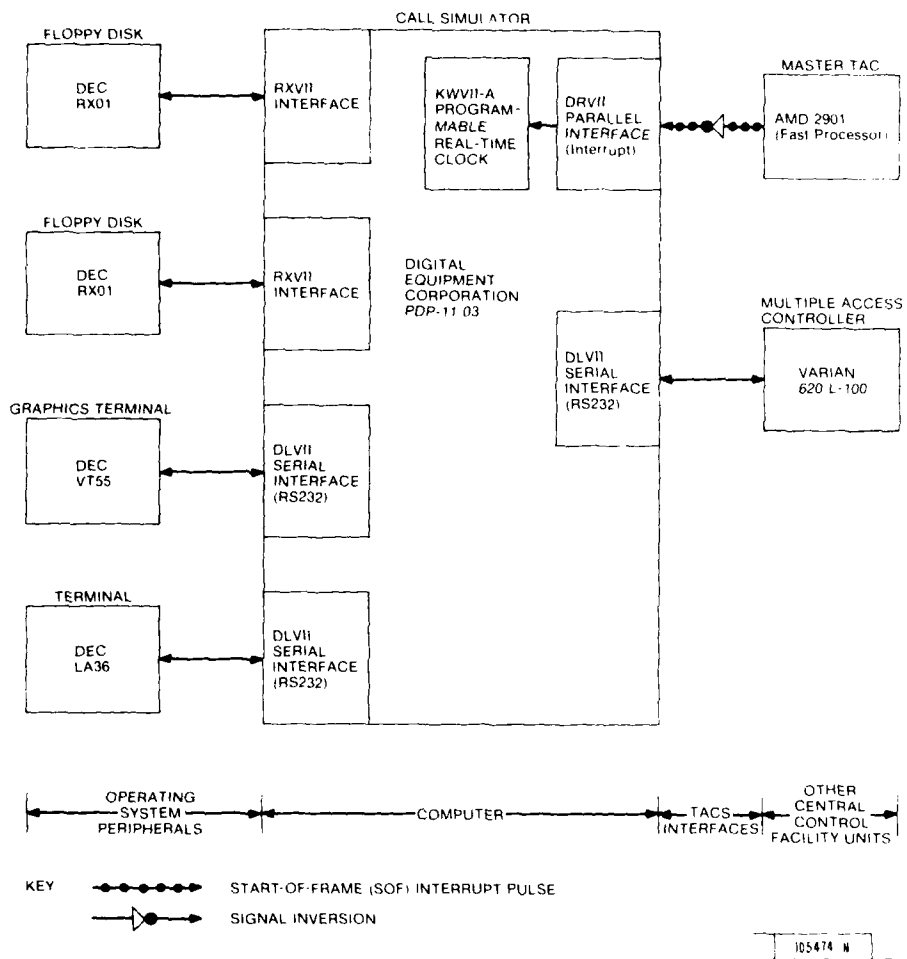


Fig. 3-2. Call Simulator hardware operating environment.

6. Conclusion

The advantages of the quite realistic Call Simulator are clear and multiple. First it allowed testing of an operational version of the MAC software by using simulated input. Thus the MAC's ability to handle requests was confirmed long before the subscriber unit development was complete. When it came time to interface the MAC and subscriber unit, it was necessary only to confirm that the TAC was formatting the requests properly and that the hardware interface was capable of transferring bits. Second, the fact that the Simulator was so realistic meant that few modifications and additions to the operational MAC programs were required in order to allow the two units to work together. This resulted in a savings in terms of MAC programming effort. Third, as mentioned previously, the compatibility in format between requests and status reports from real and simulated users means that the MAC can service both, simultaneously. This makes it possible to demonstrate, with the two real subscriber units and the Call Simulator, the grade of service that users might experience in a full-scale, fielded TACS environment. Finally, the degree to which the Simulator emulates actual subscriber units means that no special "handshaking" between Simulator and MAC is required. The Simulator simply listens to the control bursts and sends request/report bursts. It is felt that this "clean" interface lends greater credibility to both system demonstrations performed and performance data generated via simulation.

B. Implementation Details

The preceding section of this report described the functional design of the TACS Call Simulator, primarily from the viewpoints of the inputs provided it and the outputs expected of it. This section attempts to expose, in moderate detail, the way in which the Simulator was implemented on the DEC minicomputer. Included are subsections covering the hardware and software configurations. Once again, it should be noted that the material to follow pertains to the nine-slot frame structure, and that Appendix D details all significant differences for the five-slot version.

1. Hardware

The TACS Call Simulator is implemented on a Digital Equipment Corporation PDP-11/03, a 16-bit microcomputer, for this application configured with 28K words of memory. Peripherals on the system include an LA36 DECwriter terminal, a model VT55 CRT terminal, and two RX01 floppy disk drives. Figure 3-2 shows the configuration of the hardware.

Implementation of the Call Simulator requires two interrupt sources. To provide for synchronization of the entire Central Control Facility, a start of frame (SOF) pulse is generated by the Fast Processor section of the Master TAC. This pulse is connected to the signal request line of the parallel interface module (DRV11) in the DEC computer. When the signal is present, an interrupt is generated. To coordinate data transmission between the MAC and the Simulator,

a second interrupt is needed. The source of this interrupt is the programmable real-time clock (KWVII-A) present in the PDP-11/03. This clock may be programmed to generate an interrupt at any specified time from when it is enabled.

Communication between the Call Simulator and the MAC is achieved by means of a standard RS-232 serial interface between the two computers. This requires a DLV11 serial interface card in the DEC computer. The data rate for this interface is 9600 baud.

2. System Software

The operating environment of the computer is the DEC RT-11 operating system - a single job, real-time environment. The Call Simulator is linked and executed with a Digital debugging program, "ODT". The debugger allows memory to be easily examined, provides a place to proceed upon incurring an error, and allows the program to be restarted.

3. Simulation Software

The Call Simulator consists of a mixture of FORTRAN and MACRO, Digital's assembly language. The high level language is used wherever possible, with reliance on assembly language only for real-time and individual-bit-oriented operations. The software includes a non-real-time initialization segment during which the test manager is prompted to supply statistical parameters and the data base is loaded into the computer, from floppy disk. After initialization, the Simulator shifts to real-time mode, performing the same series of operations frame after frame, until the simulation run is halted by stopping the MAC. The Simulator and MAC, in conjunction, are capable of recommencing real-time operation after a temporary halt.

The general flow of data through the Call Simulator, and of the Simulator's processing sequences is shown in Fig. 3-1. The flow starts in the lower right-hand corner, with receipt of the control burst from the MAC, and forms an arch through the request generation process, ending in the lower lefthand corner, with the transmission of requests and reports to the MAC. In Fig. 3-1, single-line boxes indicate computer processing and double-line boxes signify various data storage areas (tables, buffers, etc.). The capacity, in full five-word request/reports, is shown in the corner of each segment of the request/report buffers. Following the data flow in greater detail, the Simulator first receives a control burst from the MAC, under control of its real-time and I/O routines (assembly language), placing that control information, in raw form, into the control burst buffer. The system status maintenance routines then decode and act upon information contained in the control burst. For example, the data base is updated to reflect any circuit assignments or preemptions, the circulating buffer management routines are directed to expunge all requests which have received final responses, and system configuration parameters such as the number of dedicated request/report subslots per frame are passed on to the request and report generators. Next the reports and requests for the cur-

rent frame are generated. Finally all output destined for the MAC during the current frame is assembled into the request/report output buffer, and control returns to the real-time and I/O routines which coordinate transfer to the MAC.

This report section describes each class of Simulator operation, relying heavily on the aid of flowcharts. Where possible, the flowcharts are organized in nested series, with each flowchart in the nest presenting increasingly detailed information. It is hoped that this will allow the reader to comprehend the Call Simulator algorithms to whatever depth desired.

a. Initialization Routines

The first portion of the Call Simulator software initialized parameters and computer I/O devices. Values of the average per-net call rates and durations to be used in the request generating routines are requested from the test manager and entered as input. Since requests are generated on the basis of information about TACs and nets in the system and since the MAC makes assignments also based on this information, identical information must be present in both computers. To do this, a copy of several lists comprising part of the MAC's initial data base must have been stored on a floppy disk and is read into the simulator during program initialization. Subsequent routines update these lists using control information received from the MAC. An assignment list must also be maintained in the Call Simulator; this is initialized to zero.

b. Real-time Routines

Interrupts synchronize data transfer between the MAC and the Call Simulator; that is, the transmission of the control burst from the MAC to the Simulator and the transmission of requests and/or reports from the Simulator to the MAC. An initial control burst must be received from the MAC before any request/report generation is begun by the Simulator. When the start of frame pulse interrupts, the Call Simulator enables the programmable real-time clock to generate a subsequent interrupt 1.8 s later in the frame. When this second interrupt occurs the program waits to receive the 42 eight-bit bytes of the control burst from the MAC and then waits for the next start of frame interrupt. If no control burst data have been received by that time, this sequence of events is repeated.

Once the initial control burst has been received, simulation may begin. Upon receipt of a start of frame interrupt the real-time clock is enabled. A flag is set to indicate that the current frame's processing has begun and the time is recorded. The buffer containing any requests and/or reports generated in the previous frame is transmitted to the MAC. This buffer is flexible in length to a maximum of 20 reports and/or requests. In the time before the next real-time clock interrupt occurs, processing of all the information contained in the control burst is performed as well as the generation of the next output buffer of requests and reports. At the time the next interrupt occurs, a check is made that all this processing is complete; if it is not, an error has occurred. If there is no error, the amount of time that the current frame's processing has

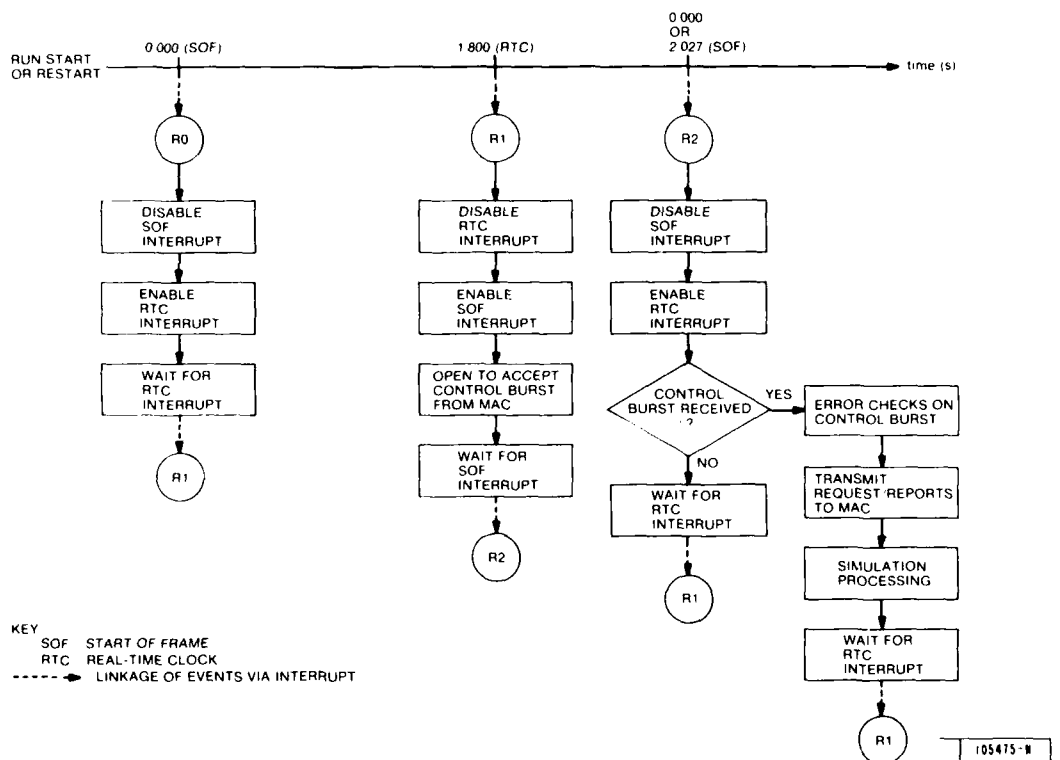


Fig. 3-3. Call Simulator real-time structure.

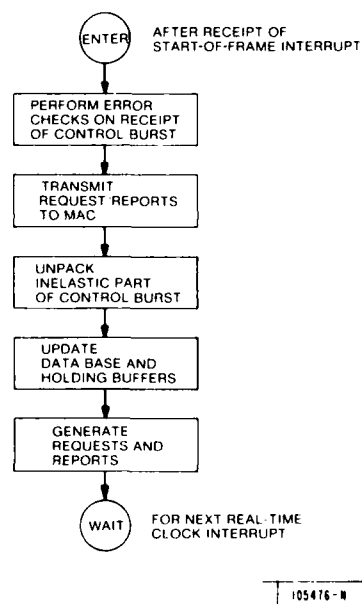
taken is computed and the maximum time over all frames to the present is stored. All of this real-time structure is programmed in assembly language. For a diagram of this structure, see Fig. 3-3.

After at least one control burst has been received, the Simulator halts at any time that the MAC stops transmitting control bursts. A message is printed on the VT55 terminal and the program branches to the debugger, ODT. It is then possible to reenter the program (manually, through and ODT GOTO command) and continue with the simulation as soon as the MAC starts to again transmit control bursts. Any requests in the output buffer are transmitted and regular processing resumes.

c. System Status Maintenance Routines

This subsection and the one which follows describe the major Call Simulator processing tasks. These tasks are performed in the time-order depicted in the general flowchart of Fig. 3-4.

Fig. 3-4. Overview of Call Simulator processing structure.



The control burst received by the Simulator is identical to the control burst received by a real TAC. It consists of both system control information and request responses. (See Appendix C for the format of the control burst.) Initially error checks are made to determine the completeness of the control burst received, as well as to ensure that the frame-of-day transmitted matches the frame-of-day count internal to the Simulator (see Fig. 3-5 for details of the handling of error conditions). Then the rest of the inelastic part of the control burst is unpacked and stored. Since the number of messages in the control burst is flexible, the message counts must be determined and recorded.

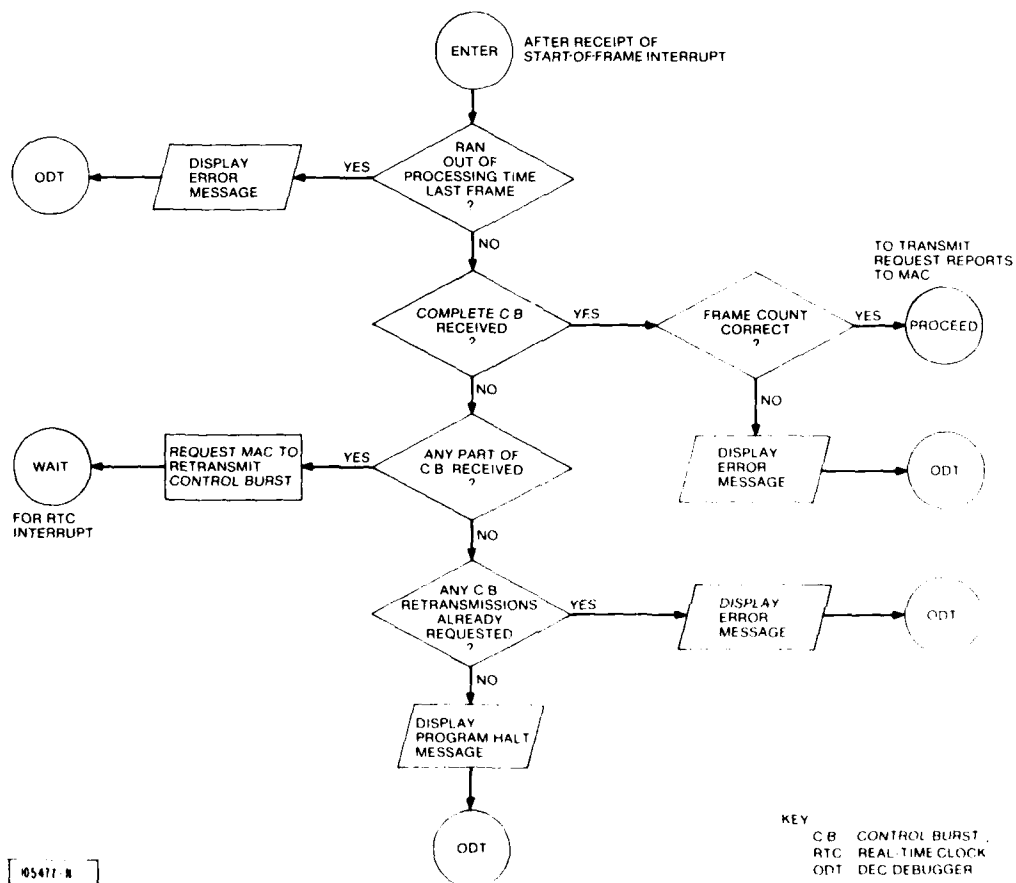


Fig. 3-5. Flowchart of error checking done on control burst receipt.

In updating the data base lists the first thing that is done is to delete all assignments whose estimated time requirement has run out. All calls generated by the Simulator have finite duration. In deleting an assignment all involved slots in the assignment table are cleared, the appropriate net list "on" bits are cleared, and the TAC list entries are zeroed. After the "timeouts" are completed, all calls which are being preempted by this control burst are deleted.

Range update, initial range, and command fields of the control burst are skipped over. As request responses are processed, all calling party addresses are compared with calling party addresses for requests being saved in the request/report or retransmit holding buffers. If a match is found, that entry is deleted from the buffer so that a request that has received a response is not retransmitted.

Information about all-member nets assigned in this frame is packed and placed in the assignment list. If the net is a voice net, the duration is converted from minutes to frames before it is stored. If the assignment is a late entry to a call already in progress, that is also recorded in the assignment list. The net list is updated by setting the bit indicating the net is on the air.

Lastly point-to-point assignments are recorded in the TAC, net, and assignment lists. Conference calls are flagged as such in the assignment list. Checks are made to see if the net is controlled. If so, all information about the net controller is also updated. Updating the data base is now complete. For a flow diagram see Fig. 3-6. The structure of the data in the assignment, TAC, and net lists is shown in Appendix E.

Initially the Call Simulator was resident in the Varian computer with the MAC, and the data base maintaining routines were part of the MAC, written in Varian assembly language. Two avenues were available for implementing this software on the PDP-11: brute force reprogramming or automatic translation of the Varian routines by the DEC assembler. The latter, automatic translation, was chosen. A special file of DEC code was written such that almost every Varian instruction was specified by a macro definition in PDP-11 assembly language. With only slight modifications, a program written in Varian assembly language can be assembled on the PDP-11 using this special file. Thus, one who closely inspects the assembly listing for some of the system status maintenance routines will find that they appear most unusual, consisting entirely of DEC macro directives!

d. Request/Report Generation Routines

When the system status maintenance routines have completed acting upon the new control burst, including updating the Simulator's copy of the MAC's data base, then the current frame's reports and requests are generated. This generation process, shown as a single block in Fig. 3-4, is indicated in somewhat more detail in Fig. 3-7, and the procedure for specifying individual

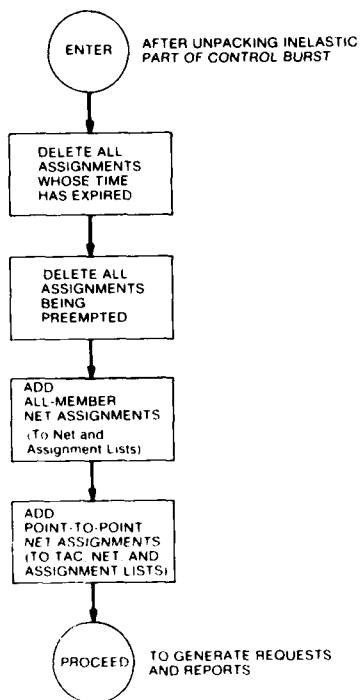


Fig. 3-6. Flowchart of Call Simulator routines for updating data base.

105478-N

request parameters is further described in Fig. 3-8. A narrative of each flowchart is presented, following the charts from the top down. Note that these flowcharts are implemented primarily in FORTRAN, with calls to a few assembly language subroutines.

Refer to the top of Fig. 3-7. At the beginning of a new time frame, the Simulator must cycle forward by one position in the request/report and retransmit holding buffers. In short, the buffer which corresponded to "frame + 1" becomes the "current frame" buffer. Usage of the buffers "circulates" in that upon finishing using the bottom buffer position, the Simulator goes back and reuses the top position. This action will happen for the retransmit holding buffers in the next frame, with the system in the state shown in Fig. 3-1. Next, the five retransmit holding buffers are searched for requests from parties whose subscriber units are about to have (dedicated) report accesses during the next three frames. When such a request is found (and if the report slot has not already been occupied by another request from that same terminal), it is allowed to anticipate the contention-free opportunity, and is moved from the retransmit holding buffer to the appropriate position in one of the request/

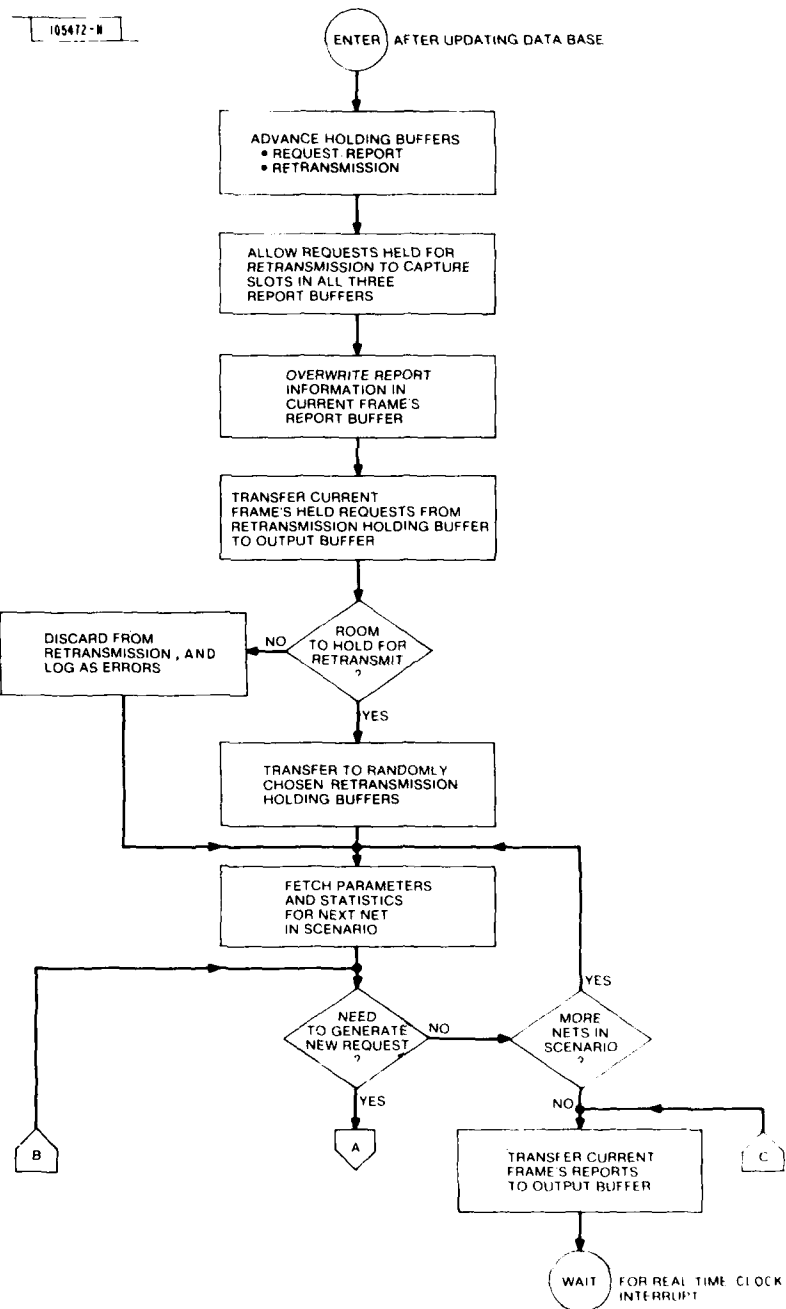


Fig. 3-7. Request/report generation, holding and sending routines.

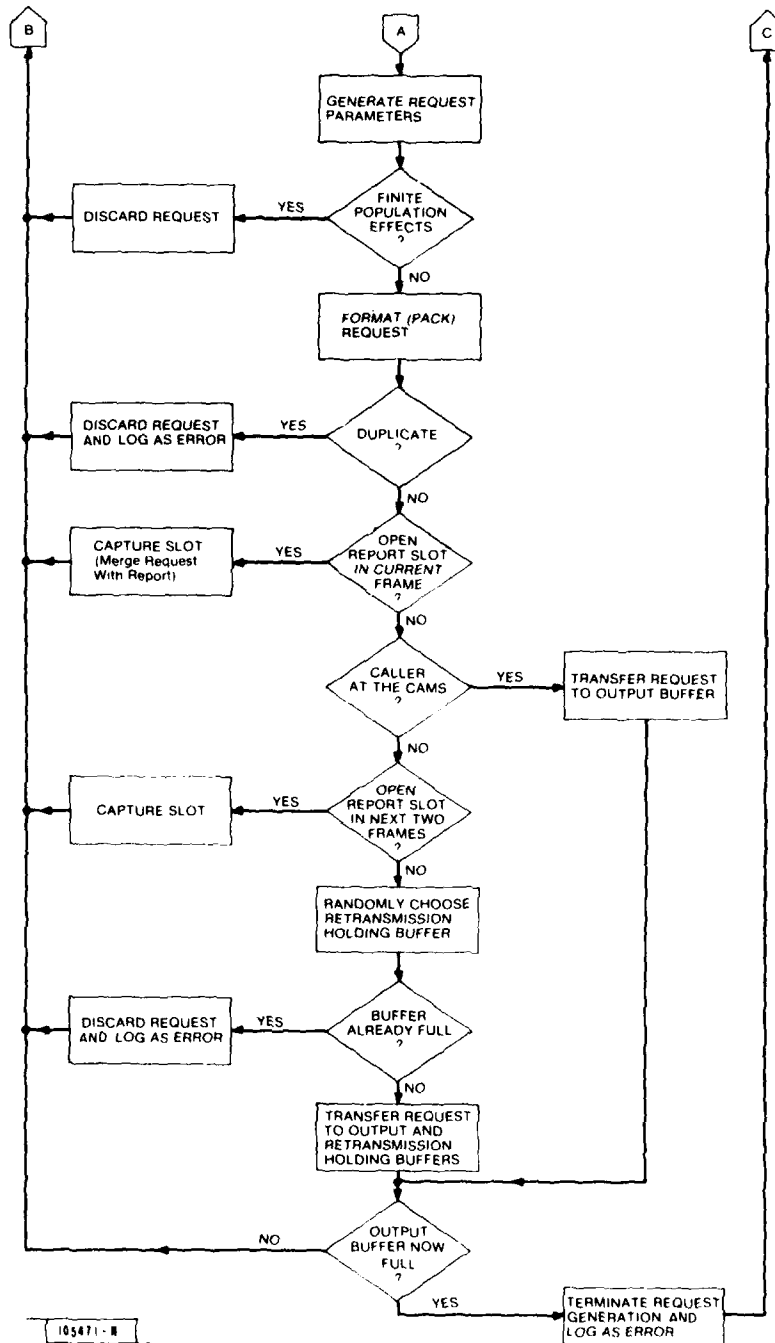


Fig. 3-7. Continued.

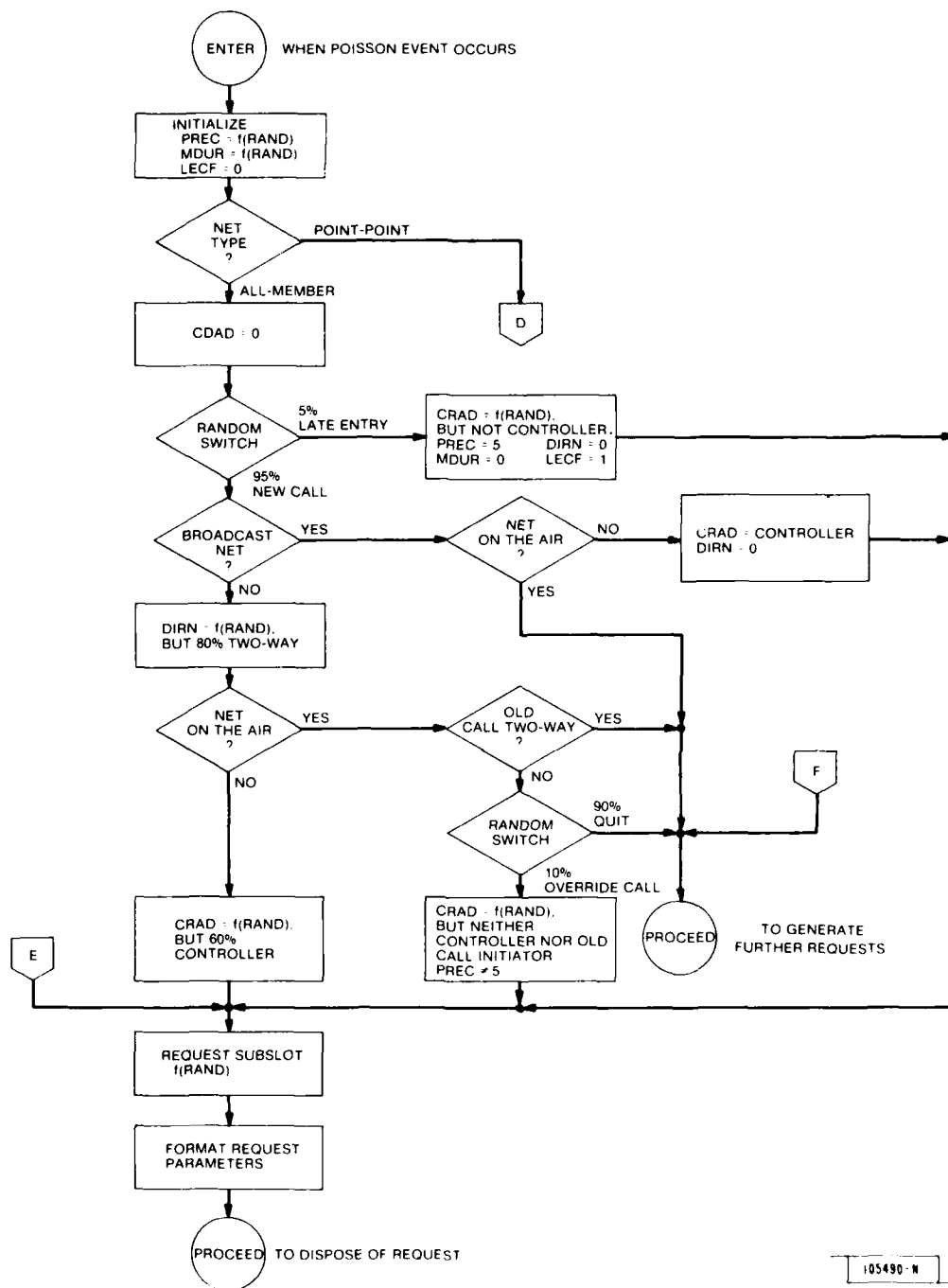
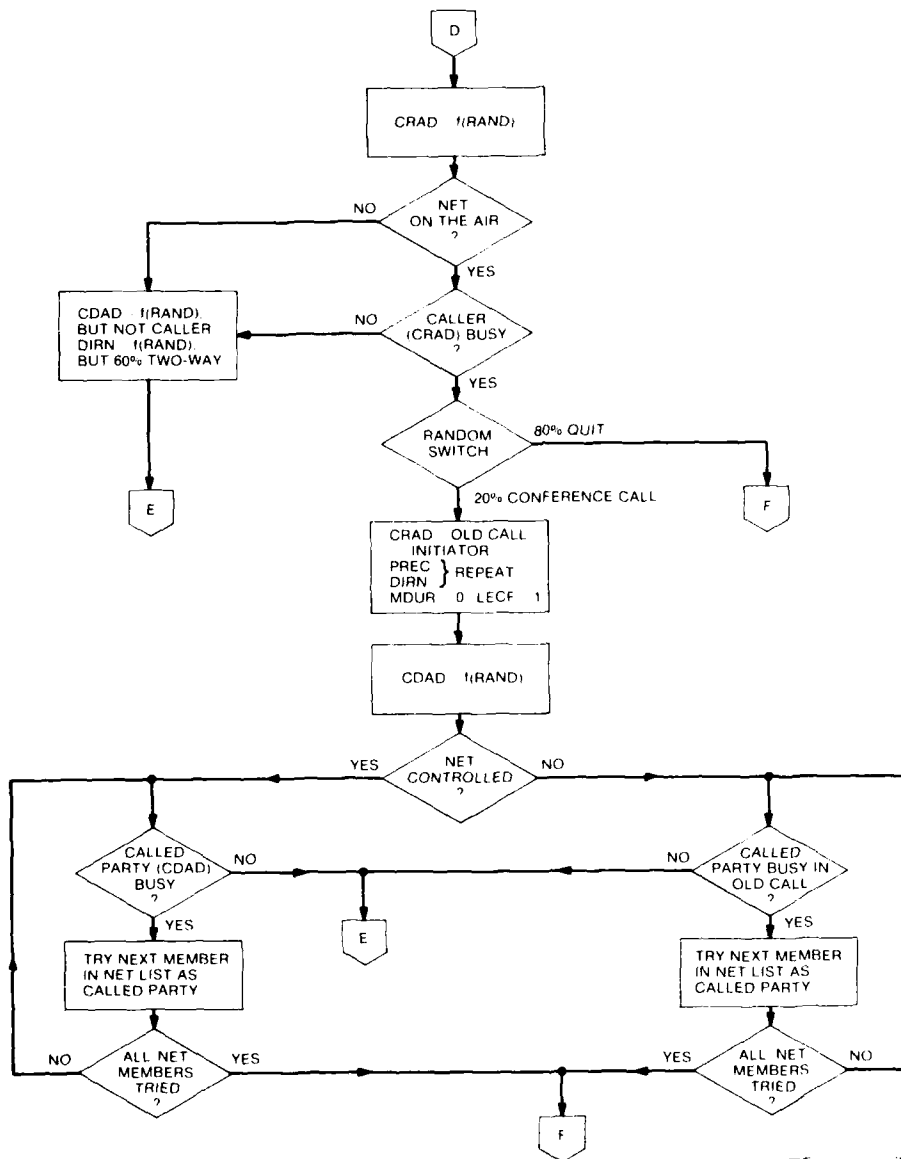


Fig. 3-8. Request parameter generation routines.



[105469-N]

Fig. 3-8. Continued.

report holding buffers. At this point, the only elements occupying the current frame's request/report holding buffer are requests which are to piggyback with the reports. The Simulator now accumulates the report information for each active terminal possessing a report slot in the current frame, and merges (overwrites) that information into the current frame request/report holding buffer along with any requests already there. This completes the composition and formatting of the current frame's request/report holding buffer. The reports (with piggyback requests, possibly), however, remain in that buffer, to be actually transferred to the output buffer later. Next, all request scheduled for retransmission in the current frame's shared request/report subslots are copied from the current frame's retransmit holding buffer to the output buffer. There is again a likelihood that these requests will suffer contention, so each one is placed back into a randomly chosen retransmit holding buffer, provided the chosen buffer is not saturated. Requests which cannot be held are lost from the retransmission scheme, but a count of these undesirable events is maintained. The new request generation process now commences, on a net-by-net basis, starting with the first net in the net list of the data base. For a given net, there is no need to generate another request in the current frame if either the specified mean call rate for the net is zero or the (pseudorandom) Poisson event generator does not happen to signal another request occurrence. Note that it is possible to have multiple events (requests) from a net in a single frame. When the Simulator finishes generating new requests from the last net in the net list, it copies the current frame's reports to the output buffer, and waits for the interrupt to coordinate data transfer to the MAC. Each time the Poisson event generator signals need for a new request, the Simulator generates a full set of pseudorandom request parameters, consistent with the present state of the system and net. That parameter generation process is more fully described in later references to flowchart 3-8. The "finite population effects" refers to the fact that already-busy users make no new requests. If the randomly chosen calling party is shown as active in the net list, the Simulator simply terminates generation of that request. The rest of flowchart 3-7 is almost self-explanatory. It has already been described in fair detail in Section III.A.4.a.

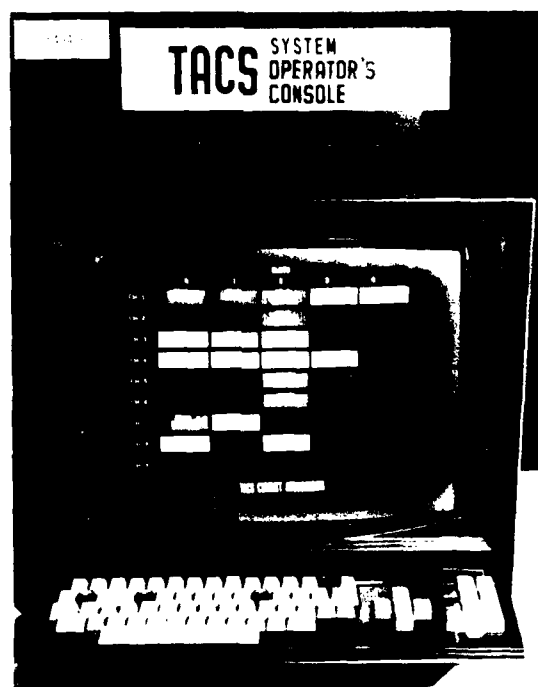
Refer to Fig. 3-8, which describes the request parameter generation process. The acronyms, with the exception of RAND, refer to individual request parameters and are defined in Table 3-1. RAND refers to the output of a uniformly distributed random number function (range: 0.0 to 1.0) named RAN, and provided by Digital Equipment Corporation, along with their RT-11 operating system version of FORTRAN. The notation: $\text{parameter} = f(\text{RAND})$, indicates a parameter which is pseudorandomly chosen, constrained to fall within an appropriate range. For example the calling party is extracted from the net list using a pointer uniformly distributed in the range $[1, N]$ where N is the number of net members. The "random switches" are also controlled, for each passage through the switch, by the output of the function RAN. Note that if

the random number function is initialized using the same "seed" and if the scenario and specified call statistics are the same for two simulation runs, then the runs will be identical, down to the last request parameter. This reproducibility for pseudorandom runs proved invaluable during Call Simulator and MAC software debugging and validation. The symmetric flowchart structure at the bottom of the branch for point-to-point nets describes the algorithm which ensures that the called party in a conference call request is not already a participant in the call. In the case of a controlled net, which can support only one call at a time, the determination is comparatively easy; if the potential called party is busy at all, he is participating in the old call.

C. Conclusion

In retrospect, there are significant ramifications of having developed our class of Call Simulator. First, upon becoming operational the Simulator saw an even greater level of usage than was anticipated, and proved itself to be an exceedingly flexible debug/test/demonstration tool. Second, implementation of the Simulator on a machine separate from the MAC required that the second machine maintain an up-to-date (in real time) copy of the MAC's data base. An alternate or back-up Multiple Access Controller would be subject to a similar requirement. Thus the successful implementation of the DEC Call Simulator has demonstrated the feasibility of having a "hot stand-by" MAC with a minimal expenditure of system overhead. The alternate MAC would merely need to monitor the system control overhead transmissions (control and request/report subslots).

1. *Journal of the American Medical Association*, 1997; 277: 1033-1038.



Journal of Management Inquiry, Vol. 17 No. 4, December 2008
DOI: 10.1177/1056492608325111
© The Author(s) 2008

This report section describes the System Operator's Console functions and their implementation. Section IV.B describes the characteristics of the Intecolor 8051 terminal used for the Operator's Console. Section IV.C summarizes the design philosophy of the console software. Section IV.D provides a detailed description of the display package and the operator commands. Sections IV.E-IV.H provide a more detailed description of the display software.

B. The Intecolor 8051 Intelligent Terminal

A color CRT was selected for the System Operator's Console for a number of reasons. First, sufficient advances in color technology have been made such that reasonably priced units are available. Second, color graphics offers significant advantages over the traditional black and white displays: Color allows emphasis of critical message content, more attractive and comprehensible message presentation, and reduced operator fatigue when monitoring the display. These benefits were evidenced in the very active use of the display terminal during the later stages of the development cycle of TACS.

The Intecolor 8051 terminal is a low-cost, intelligent, RAM-refreshed, eight-color (including black and white) CRT. The terminal contains a programmable Intel 8080 CPU and up to 20K bytes of user RAM. A pair of mini-floppy diskettes handled by a manufacturer-provided disk operating system allows reasonable flexibility in program development. A standard RS-232 port is provided which serves as the interface to the Varian minicomputer. Alternatively, a printer may be attached to this port.

The terminal has a 19-in. CRT and a 48-line by 80-character alphanumeric display capability. The graphics mode allows points, vectors, and bars to be generated. The graphics resolution is 160 x 192 units. Color may be selected for either the foreground or the background in all plotting and character display modes.

The terminal supports programming in BASIC with an interpreter and Intel 8080 assembly language with choice of a disk or a ROM assembler.

C. Software Design Philosophy

The System Operator's Console provides the Multiple Access Controller with a system status display facility as well as an operator command capability. Because the terminal is an intelligent one, display-oriented processing of the appropriate section of the MAC data base is done by the terminal, thus distributing the total processing load. For that reason, a significant part of the Multiple Access Controller data base (the "system status information") is transferred to the Intecolor 8051 terminal every frame. The Intecolor terminal then reduces the data to a form appropriate for display. All of the requisite data manipulations are done in real time by the terminal so as to provide a responsive system.

The program in the Intecolor 8051 was designed for simplicity of use by the system operator and for maximum efficiency and reliability in providing both a selection of displays, a command mode, and the capability of dynamically changing between them. The program occupies 12K bytes of RAM, approximately 60% of the total amount of RAM space available in the Intecolor terminal.

1. Interface to the Multiple Access Controller

The Multiple Access Controller is synchronized to a time frame of approximately 2 s. Thus, every 2 s, new circuit assignments may be made and old ones terminated. In order for the System Operator's Console to accurately and usefully depict the status of the MAC, it must receive data from the MAC each frame. The data transferred consists principally of the MAC assignment list (a list of all time slots in all frequency channels with data regarding the disposition of each slot), the request queue and various other data including frame count.

As no DMA capability is available within the Intecolor terminal, programmed input/output was the means by which the data transfer to and from the Varian minicomputer was accomplished. Further, as the interrupts from the RS-232 interface are serviced by the CRT software, the data could not be received on an interrupt basis. Thus, the program was written to perform a display cycle and then to proceed to a wait loop until the next burst of data is sent by the MAC computer.

2. Choice of Programming Language

Because of the non-interrupt nature of the data transfer and also because of the desirability of presenting up-to-date and consistent displays, it was important that all display processing finish within the two second frame. For this reason, the BASIC interpreter was discarded as a possibility for the display program despite the inherent desirability of programming in a higher level language. The programming was therefore done in Intel 8080 assembly language. Due to the size of the program, the source code development was not done on the Intecolor 8051 but on the Lincoln Laboratory central computing facility, an IBM 370/168 computer. The machine code was then downloaded to the terminal.

3. Program Structure Considerations

The Operator's Console display program was designed to function in such a way that the operator can dynamically change the display or enter the command mode. In order to assure quick response time, the entire package of displays and commands was written as a single program so that disk accesses would not be necessary. Sufficient RAM space was available for the entire package; the program structure was therefore straightforward.

In order to achieve maximum flexibility, the displays were designed to be interruptible by the operator at any moment. Because of this requirement and the desirability of continuously updating the display, polling of the Intecolor keyboard is done once per frame. This polling is made possible by the use of an operating system utility routine. The operator must therefore depress the "attention key" for a maximum of one frame (2 s) before the display program recognizes the operator's signal.

Once the operator has chosen a display, it is desirable to have the selected display appear on the CRT screen with minimum delay. The program therefore does not wait for the next burst of data from the MAC; rather it immediately proceeds to the prescribed display generation code. As a result, the response time of the terminal between displays is virtually immediate.

D. TACS Display and Command Package

Six displays and a command mode have been implemented for the System Operator's Console. The displays consist of a frame structure display, a request queue display, system statistics, details on an individual time slot assignment, and bar graphs of satellite capacity utilization and of the precedence distribution of assigned slots. In addition, the command mode offers the operator three capabilities: deletion of a frequency channel from the demand assignment pool, addition of a channel to that pool, and the preemption of any single assignment.

TACS has been developed to support two frame structures, with frames consisting of either five or, for greater efficiency, nine slots. The System Operator's Console program is capable of generating displays for either frame format. At system startup time, the operator must specify which frame structure is in use.

Once the operator has entered the number of slots per frame, the program waits for a burst of system status data from the MAC. After receipt of a complete data burst, the frame structure display will appear. This display provides an overview of the current allocation of satellite resources and is dynamically updated each frame. The operator may, however, wish to view another display or to enter the command mode. In order to change modes, the "?" (attention) key must be depressed for no more than 2 s until the "menu" appears on the screen. The display menu enumerates all of the Operator's Console displays. To select a display or to enter the command mode, the operator must enter a digit between 1 and 7. The selected display will then appear on the screen. Figure 4-2 depicts the menu format.

In the following sections, each of the TACS displays and the command mode are discussed. More detail regarding the programming of each is found in Sections IV.E-IV.H.

[105408-N]

TACS DISPLAY SYSTEM

SELECT COMMAND OR DISPLAY

- (1) FRAME STRUCTURE
- (2) REQUEST QUEUE
- (3) SYSTEM STATISTICS
- (4) DETAILED SLOT DATA
- (5) SATELLITE CAPACITY UTILIZATION
- (6) PRECEDENCE DISTRIBUTION
- (7) COMMAND MODE

Fig. 4-2. Display menu.

1. Frame Structure Display

This display is a grid composed of boxes for each time slot in each frequency channel. Figure 4-3 shows an example of this display for the nine-slot frame. The utilization of each slot is recorded within the appropriate box. If no call has been assigned to the slot, it remains blank. If the slot is being used for system overhead, that is, for request/report subslots or control and ranging subslots, the slot is marked accordingly.

If a slot contains a circuit assignment, the block will have its background color set to correspond to the precedence level of the assignment. As TACS honors calls of five precedence levels, five of the eight CRT colors are used to indicate the assignment precedence. The most eye-catching colors are assigned to the higher precedence assignments. Within the block, three lines of ASCII data are recorded. The first line expresses the precedence explicitly. The second line contains the net address; the third line displays the calling and called TAC unit addresses if the call is from a point-to-point net. For an all-member net, the third line is left blank.

The display is updated dynamically each frame. Therefore, new entries are continuously added and old entries deleted to reflect the MAC's circuit assignments and terminations. Further, at times, calls must be preempted if higher precedence calls require resources which are used by lower precedence calls. Preemption can occur for a variety of reasons; for example, if satellite capacity is not available or if an I/O port is called by a party with a message of higher precedence than the call currently in service. When an assignment is preempted, the corresponding block in the frame structure grid is colored red, and "PREEMPT" flashes within the block, thus calling the operator's attention to the event.

The operator also has the ability to remove channels from the demand assignment pool. When a channel has been removed, no subsequent assignments will appear in that channel and all circuits occupying it will be preempted. As a confirmation that the channel has indeed been removed from the pool, a red asterisk is placed next to the channel number on the left side of the display.

	SLOTS								
	0	1	2	3	4	5	6	7	8
CH 1	PR-5 NET-003 063/063	PR-5 NET-004 063/063	PR-5 NET-002 063/063	PR-5 NET-005 063/063			PR-5 NET-012 063/063	PR-5 NET-007 063/063	PR-5 NET-001 063/063
CH 2	CONTROL		REQ/REP	PR-5 NET-016 063/063		PR-5 NET-018 063/063			PREEMPT
CH 3			REQUEST	PR-5 NET-015 063/063		PR-5 NET-007 063/063			
CH 4									
CH 5									
CH 6									
CH 7			PR-5 NET-004 103/063	PR-5 NET-015 103/063		PR-5 NET-014 103/063		PR-5 NET-010 103/063	
CH 8					PR-5 NET-015 103/063	PR-5 NET-016 103/063	CONTROL		REQ/REP
CH 9									

TACS CIRCUIT ASSIGNMENTS

4

2. Request Queue Display

The Request Queue display shows the contents of the Multiple Access Controller request queue each frame. The MAC request queue contains all currently pending subscriber unit circuit requests. By providing a dynamic view of these requests, the display package permits the operator at the TACS Central Control Facility to monitor the performance of the system - particularly as regards the number and type of request blockages occurring.

The screen shows an ASCII representation of a selected subset of request parameters as well as the number of frames the request has been in queue and the disposition of each request as of the current frame. It should be noted that the queue includes both newly arrived and previously held over requests. For all cases, information regarding the type of call and the calling party's net and TAC address is displayed.

Figure 4-4 shows the format of the display and an example of the MAC queue contents. In the first column is the call precedence, which ranges from 1 to 5. The queue is sorted by precedence; therefore, the lowest numbered (highest priority) requests are listed first. Next are the net address and the calling party's TAC address. Following is the type of circuit requested: two-way or broadcast. The number of request attempts is then indicated. Requests by subscriber units are made by randomly selecting a request/report subslot and then broadcasting the request. If two units simultaneously broadcast a request in the same subslot, neither will arrive at the MAC due to RF interference. Consequently, the subscriber units are programmed to retransmit requests periodically if no response has been received. The number of request attempts is a count of the number of transmissions required before the request reaches the MAC and provides a measure of the amount of contention for the random access request/report subslots.

"Frames in queue" reflects the number of frames that the request has so far remained in the MAC queue pending final action. In the last column, the action taken on the request is shown. That action may consist of an assignment; in that case, the assignment will be either be a point-to-point or an all-member assignment, depending on the type of net requesting service. The action taken may also consist of a blockage; in that case the message shown will correspond to the call progress message displayed on the calling party's TIC. Figure 4-4 shows the case of a request that has been blocked due to "terminal busy" conditions. Requests that will be held over in queue are flagged as "left in queue" in the action column.

The request queue display is updated each frame, and observation of the display by a system operator provides useful information about the current blockage rate and types. If the system is overloaded, the operator may see the queue become saturated, changing from a few requests per frame to 10-20 or more. As the Intecolor screen can hold only 20 queue elements comfortably, queue

CP202-28890

MULTIPLE ACCESS CONTROLLER REQUEST QUEUE

PR	NET	CALLER	TYPE	ATTEMPTS	FRAMES IN QUEUE	ACTION
2	018	109	TWO WAY	0	00	PT-TO-PT ASSIGN
4	015	060	BROADCAST	0	01	TERMINAL BUSY
4	020	063	BROADCAST	0	01	PT-TO-PT ASSIGN
5	001	042	BROADCAST	0	00	ALL MEMBER ASSIGN

Fig. 4-4 MAC request queue display

CP202-28890

TACS SYSTEM STATISTICS

FOR

THE NINE SLOT FRAME

SYSTEM PERFORMANCE

34% CAPACITY IN USE
 23 CALLS IN SERVICE
 00 ALL MEMBER CALLS
 15 PT-TO-PT CALLS
 AVERAGE WAIT TIME= 07 SEC

LINK QUALITY

00 CALLS BLOCKED FOR
 INSUFFICIENT LINK
 22 EXCELLENT LINKS
 00 GOOD LINKS
 01 FAIR LINKS

BLOCKAGES

01 CALLS PREEMPTED
 03 REQUESTS IN QUEUE
 02 ASSIGNED
 01 BLOCKED
 00 WAITING

BLOCKAGE TYPES

01 TERMINAL BUSY
 00 SATELLITE BUSY
 00 TERMINAL OFF
 00 INSUFFICIENT LINK
 00 NET INACTIVE
 00 NET ALREADY ACTIVE
 00 INVALID REQUEST

entries beyond the twentieth do not appear. As queues of that length are indicative of system saturation, no attempt was made to extend the display.

The queue is primarily lettered in yellow on a black background. The heading information is depicted in white.

3. System Statistics Display

The System Statistics display summarizes the current state of the TAC system. Data are presented that provide an overview of the activity taking place within the Multiple Access Controller during the current frame. Data regarding the number and kinds of circuits assigned, the processing of requests, the link quality for assigned circuits, and the types of blockage encountered by requests processed during the current frame are presented. The display is updated each frame.

Figure 4-5 shows an example of the System Statistics display. In the upper left box, system performance data are recorded. The current percentage of satellite capacity utilized is listed. That calculation is done for the current frame only. It reflects the capacity occupied by both user data circuits and the overhead required for system management. The number and types of calls in service are listed next. If the calls require multiple slots, as they do when link quality is suboptimal, a single call may represent the utilization of more than one time slot. The average wait time for a call assigned by the MAC during this frame is shown next. This quantity reflects the amount of time spent waiting in the MAC queue for final action, the amount of time spent in retransmission of the request and a 4- to 6-s minimum for routing of system data.

The upper right box summarizes the preemptions and the dispositions of the requests in the MAC's queue for the current frame. First, the number of assignments preempted as a result of servicing this frame's requests is listed. Next is the number of requests in the MAC's queue at the start of the frame. Their disposition is displayed in three categories: assignment, blockage, and waiting in queue.

In the lower right box, more detail is presented regarding call blockage. The number of blockages of each of the following types is listed: terminal busy, satellite busy, terminal off, insufficient link, net inactive, net already active, and invalid request.

The lower left box of the display gives an overview of the link quality of the active subscriber community. First listed is the number of calls blocked this frame due to link quality so poor that the circuit could not be assigned. Next is a summary of the link qualities of the currently assigned circuits. For example, for the five-slot frame structure case an excellent link entry indicates that the circuit is functioning at a burst rate of 19.2 kbps with a code rate of 3/4. The "good" links are transmitting and receiving at a burst rate of 19.2 kbps and a code rate of 1/2; the "fair" links are using

CIRCUIT ASSIGNMENT

FOR

CHANNEL 1 - SLOT 2

TYPE OF CALL

CALL PRECEDENCE =

SLOT(S)

TWO WAY

POINT TO POINT NET

ADDRESSES

NET =

CALLER TAC =

CALLEE TAC =

FRAME COUNTS

REMAINING =

CURRENT =

INITIAL =

RATES

BURST =

CODE =

a burst rate of 9.6 ksps and a code rate of 1/2.

The use of color in this display is primarily to highlight the data and to outline the blocks in such a way that the operator can absorb the large amount of data presented. The background is dark blue, and the lettering is in a lighter color. The numbers are in white, which provides maximum contrast.

4. Detailed Slot Data Display

This display allows the system operator to view all of the data maintained by the Multiple Access Controller for a given time slot in a given frequency channel. When this display is selected, the operator is requested to input the channel and slot number of interest. That prompting sequence is shown in Fig. 4-6. Once these values have been obtained, the characteristics of any call occupying the specified slot are displayed as shown in Fig. 4-7. Four categories of data are presented: type of call, addresses, frame counts, and rates.

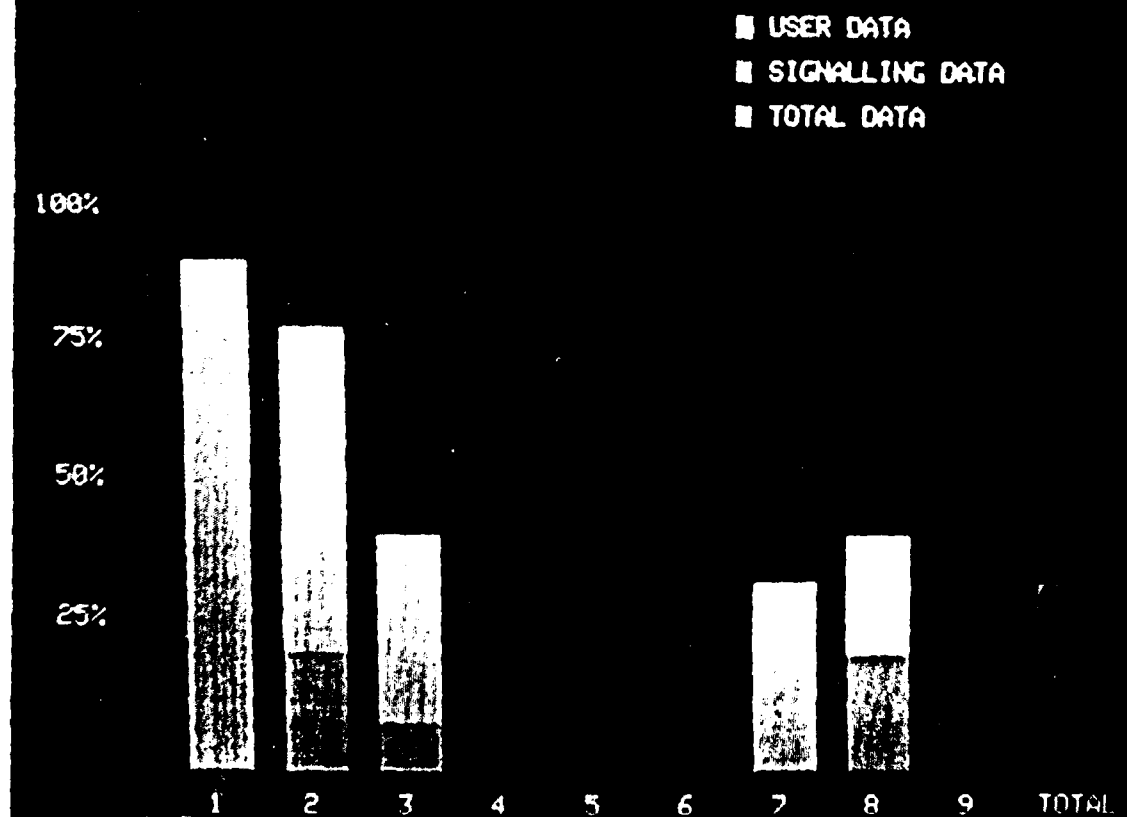
If the slot contains a circuit assignment, all four of the boxes are filled with data describing the nature of the call assigned to that slot. If the slot is not occupied, a single message, "slot available," is displayed in the box labelled "type of call." If the slot is being used for system overhead, the type of overhead is identified - e.g., "control slot." If a call has just been preempted, "preempted" appears in red as the sole entry in the "type of call" box. The display is updated each frame; therefore, a slot that is vacated and subsequently reassigned will pass through a number of states within a few frames.

Figure 4-7 is an example of the detailed slot data display for a point-to-point call occupying channel 1, slot 2. Under "type of call," it may be seen that the call precedence is 1, that the call occupies a single slot, that it is a two-way call (as opposed to a broadcast one), and that a point-to-point net is on the air. If this were a conference call, an additional line ("conference call") would appear in this box.

The addresses of the parties to the call and the net address are provided in the upper right box. In the lower left box are the frame counts. The MAC requires an estimate of call duration when the call is placed. If the duration is other than indefinite, the call will be terminated after the specified number of frames. This section of the display provides the data regarding the remaining duration of the call. Next is the current frame-of-day count from the MAC. Finally, the MAC frame-of-day count when the circuit was assigned is shown.

In the lower right box are the burst and code rates for the circuit. For the five-slot frame (of which this display is an illustration), the nominal burst rate is 19.2 ksps and the code rate is 3/4. In the case of degradations in link quality, the burst rate may decline to 9.6 ksps and the code rate to 1/2.

SATELLITE CAPACITY UTILIZATION



The display is shown on a black background as it offers maximum contrast with the majority of the colors available on the terminal. The colors used for lettering and numbers are primarily green, white, and yellow, those which are easiest to read on a black background.

5. Satellite Capacity Utilization Display

The satellite capacity utilization bar graph shows the current percentage of satellite transponder capacity in use by the TACS system. The bar graph is generated each frame and gives an overall picture of the extent and way in which satellite resources are managed by the MAC.

The data are first presented for each of the nine frequency channels potentially managed by the MAC. The amount of capacity in use, which is quantized by slots, is shown for both system overhead (signalling, or orderwire) and user data circuits. The signalling data bars are in light blue and as a rule change very little over time. The user data bars are yellow; they grow and shrink frequently as circuits are assigned and terminated. Figure 4-8 shows an example of this bar graph. Note that the majority of the system overhead is derived from channels 2 and 8 with a smaller amount in channel 3.

The total percentage of satellite capacity used is presented in the bar labelled "total," which is graphed in violet. That bar grows and shrinks dynamically but much more slowly than the user data bars and thereby provides a reasonably stable measure of the consumption of satellite resources by the TACS user community.

6. Call Precedence Distribution Display

The distribution of call precedence for the currently assigned slots is shown in this bar graph, which is illustrated in Fig. 4-9. Each precedence is shown in a different color on a black background. The colors are the same as those used in the frame structure display. The bar graph is dynamically updated so that the bars may grow or shrink each frame. This display provides the operator with one measure of the system's performance. For example, when the system saturates, the percentage of precedence 1 assignments dramatically increases.

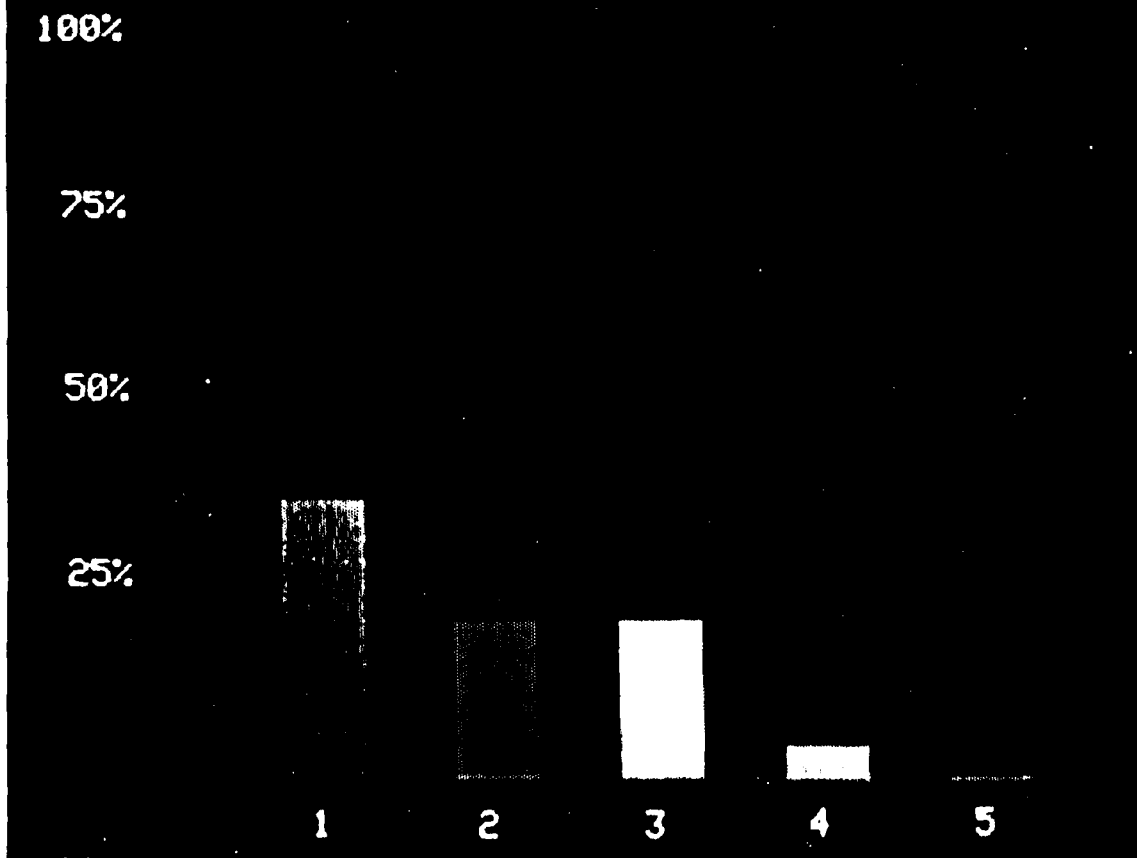
7. Command Mode

When the command mode is selected, the System Operator's Console prompts the operator to enter a command. Three commands are recognized: the first allows the deletion of a frequency channel from the demand assignment pool, the second allows the addition of a channel to the pool, the third preempts the assignment in the time and frequency slot specified.

The Operator's Console performs validity checking on the operator command parameters before sending the command to the MAC. If the program finds an error in the input, another command is requested from the operator. The "ESCAPE" (abbreviated to "E" if desired) command will cause the terminal to return to the

REF ID: A7941

CALL PRECEDENCE DISTRIBUTION OF ASSIGNED SLOTS



100% 75% 50% 25%

1

2

3

4

5

display menu for another selection. All commands must be terminated by a carriage return. Once a command has been sent to the MAC, the Intecolor terminal returns to the frame structure display so that the operator may see the effect of the command.

- a. Deleting a frequency channel from the demand assignment pool.

This command is issued by entering the following on the Intecolor keyboard:

DELETE n

where "n" is a frequency channel number between 1 and 9. The frequency channel number is checked for validity. It must be between 1 and 9 and the channel must be currently in the MAC demand assignment pool. Further, the console checks to see if any slots in that channel are being used for control or request/report subslots. If so, the deletion is not permitted and the operator is informed of this fact.

This command may be abbreviated by omitting any or all of the letters following the "D". This abbreviation allows the operator who is familiar with the system to enter the command quickly. Also, the blank between the command and the channel number is optional. Therefore, the most expeditious means of entering a command to delete channel 3 is the following: D3.

- b. Adding a frequency channel to the demand assignment pool.

Invoking this command requires entering

ADD n

on the Intecolor keyboard where n is the number (from 1-9) of the frequency channel to be added to the demand assignment pool. Again, the channel number is checked for validity. The range of the number as well as the channel's out-of-pool status is checked. Should either check fail, the operator is so informed, and an additional command is requested.

As in the case of the deletion command, abbreviation is permitted. "A4" is the shortest means of entering a command to add channel 4 while "ADD 4" is the longest.

- c. Preempting an assignment.

The system operator is empowered to preempt individual assignments. To do so, s/he must enter the following at the Intecolor terminal:

PREEMPT CH n SL m

The issuing of the above command with a frequency channel (1-9) substitution for n and a slot number (0-8 for the nine-slot case, 0-4 for the five-slot case) for m will result in the preemption of the assignment occupying that slot. Validity checking is done on the data received. In addition, the latest copy of the MAC

AD-A102 928

MASSACHUSETTS INST OF TECH LEXINGTON LINCOLN LAB
TACS CENTRAL CONTROL FACILITY.(U)

F/G 17/2.1

FEB 81 S B STORCH, L E TAYLOR, S N LONDON

F19628-80-C-0002

UNCLASSIFIED

TR-542

ESD-TR-81-1

NL

2 OF 2

AD-A
102928

END

DATE

FILED

9-81

DTIC

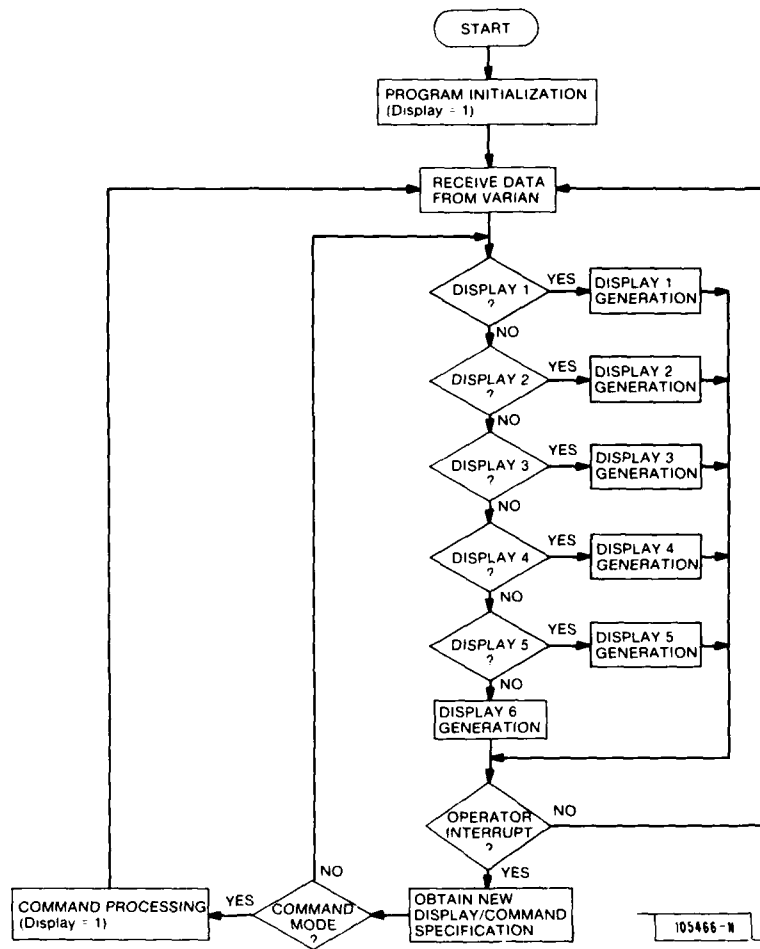


Fig. 4-10. System operator's console program design.

assignment list is checked to ensure that a call is indeed occupying that slot. If there is none, the preemption command is not forwarded to the MAC, and the operator is given the opportunity to enter another command. If a call does occupy the specified slot, the net number of the call to be preempted is displayed before the terminal returns to the frame structure display.

As with the channel deletion and addition commands, abbreviation is permitted. "P56" is sufficient to command the termination of the call occupying channel 5 and slot 6 in a nine-slot frame structure.

E. Detailed Software Description

1. Overall Program Design

The overall structure of the System Operator's Console program is simple and straightforward, a design which has contributed to the program's reliability and the ease of adding displays and commands to the package. After program initialization, data are received from the MAC in the Varian computer. Once the data have been received, the program goes on to determine which display is currently selected. That display is then updated for the current frame. Before returning to receive the next frame's data from the MAC, the program checks to see if the operator is signalling (depressing the "?" key). If so, the display menu is provided and the display/command mode selector flag is changed according to operator input. The program then proceeds back to receive data from the MAC; it continues on in a similar manner frame after frame. Figure 4-10 illustrates this design in flowchart form.

2. Program Initialization

At this time, a number of one-time initialization operations are performed. The most important of these is obtaining input from the operator as to whether the displays are to be generated for the case of a five- or nine-slot frame. The operator is prompted as, "NUMBER OF SLOTS = "; when a "5" or a "9" is received, the program stores that information and proceeds. At this time, the display/command selector flag is set to one, selecting the frame structure display.

3. Receiving Data from the Multiple Access Controller

Next and on each cycle, system status data are received from the MAC in the Varian minicomputer. The data stream consists of a pair of header bytes (ASCII '/') on which the Operator's Console synchronizes itself, the MAC assignment list, MAC request queue information, the frame of day, information on ranging/report/request subslot activity, and a number of spare bytes. Because the length of the assignment list differs for the five- and the nine-slot case, the amount of data received by the Intecolor terminal also differs in these two cases. For the nine-slot case, 758 bytes are received each frame; while for the five-slot case, 470 bytes are received. Details of the data stream format may be found in Appendices C and E.

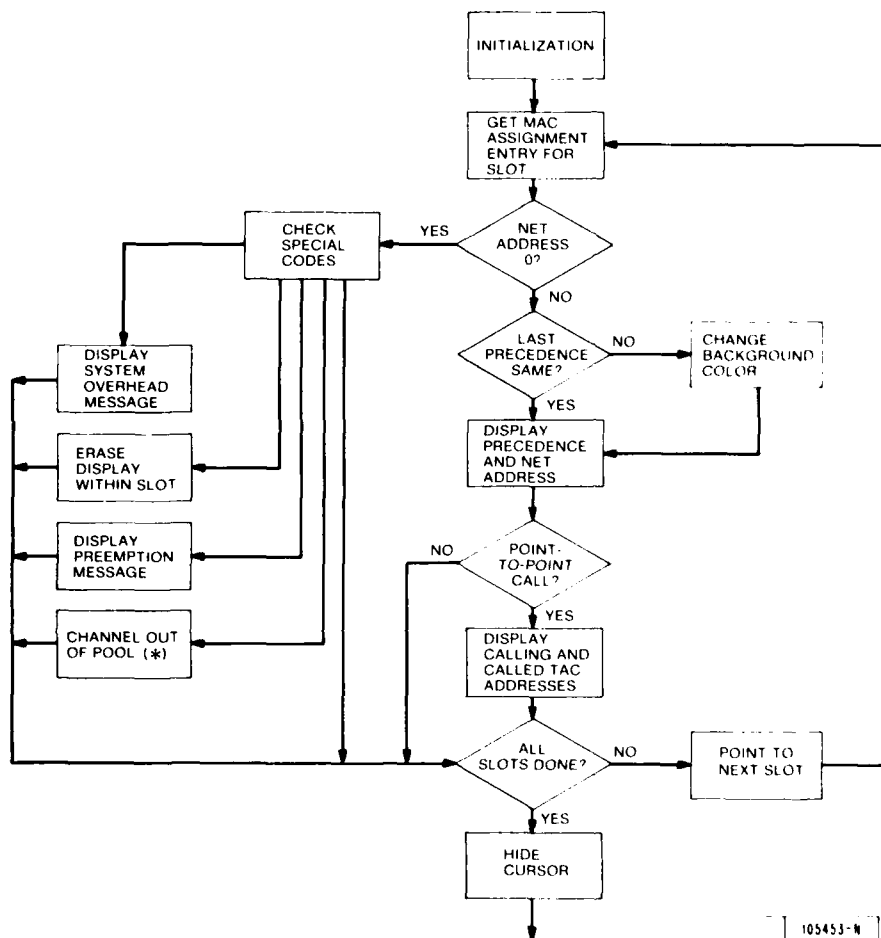


Fig. 4-11. Frame structure display flowchart.

The data are received by programmed input/output. Therefore, if the program is unable to complete a cycle in time to catch the header bytes from the MAC, it must wait until the next cycle in order to update its displays. In practice, the program executes sufficiently rapidly that this does not occur except during grid generation for the frame structure display, a phenomenon that only occurs when the display is first selected.

As the data are received from the Varian minicomputer, they are stored into RAM for the use of the display routines. No data are retained from previous cycles; hence there is no buffering. Once all of the data have been received, the program proceeds to update the current display.

4. Frame Structure Display

A flowchart of this program segment is shown in Fig. 4-11. The following sections discuss the structure laid out there.

a. Initialization

On the first cycle when this display has been selected, the old display is erased, and the part of the display which will remain fixed is generated. That is, the frame structure grid and the labels are displayed. The grid contains 45 (9 channels by 5 slots) boxes for the 5-slot case and 81 (9 channels by 9 slots) for the 9-slot case.

As there are no general-purpose graphics available for the terminal, the grid generation is done using the hardware graphics capability directly, and the ASCII labelling is done by using simple low-level CRT operating system utilities which interpret either a single character or a string and correspondingly alter the RAM refresh memory for the CRT. Details on this procedure are provided in Section IV.F.

b. Slot Usage Display

The slot usage code is executed on every frame in which the frame structure display is selected. This code cycles through the MAC assignment list and enters data into those slots which contain entries. Because it is time-consuming to send display data to each slot, the program only updates the display for those slots which have significantly changed status in the current cycle. That is, if a slot is assigned or newly relinquished, the display is updated. Otherwise, the slot is skipped insofar as the display hardware is concerned.

Because this updating procedure is selective, the display code must retain the status of every slot from the last frame. For that reason, this display routine saves a shortened version of the MAC assignment list for one frame in RAM so that the display may be intelligently updated. For assigned calls, the precedence is saved so that the colored background, which is coded by precedence, need not be generated again; this precaution is necessary in order to prevent the display from flickering each frame. For slots used for system overhead (signalling), a numerical code corresponding to the type of overhead usage

is saved.

As the program cycles through the assignment list, the net address field of the assignment list entry is first checked. Figure E-1 shows the format of a MAC assignment list block. If the net address field is non-zero, this time slot is assigned to a net. If this is the case, the program proceeds to obtain the precedence of the call occupying this slot on the last cycle. If the two precedences match, the background color block generation is skipped. If they do not match, a block of color is generated which will serve as the background for the ASCII data concerning the call. The colors are selected by precedence and are assigned as follows:

<u>Precedence</u>	<u>Color</u>
1	Violet
2	Green
3	Yellow
4	Light Blue
5	Dark Blue

The brightest and most eye-catching colors are assigned to the lowest numbered (highest) precedence calls. Red is reserved for preemptions. If no call is present in the slot during this frame and yet a call occupied the slot in the last frame, the background is colored black so as to erase the old entry from the screen.

Once the block of background color has been displayed, the precedence is displayed explicitly in ASCII in the top line of the block. Entries of the form "PR=n" therefore appear in each slot which has a circuit assigned to it. The net address is decoded next and displayed in the second line of the block. The address appears as follows: NET=nnn.

For the case of a point-to-point net, two further quantities are displayed. These quantities are the calling and called TAC addresses. They are displayed on the third and final line within the slot's block. If TAC 005 were calling TAC 124, the entry would appear as 005/124. For all-member nets, this text line is not used.

If the net address for a given assignment list entry is found to be zero (i.e., if the slot is not assigned to a net), the low order byte of the frame of assignment field is checked. That byte is used by the MAC as a means of transferring additional information to the System Operator's Console. Specifically, there are special codes that may be found in this byte. They may indicate that the slot is available for assignment, that it is in use for system overhead, that the channel in question has been removed from the demand assignment pool or that the call occupying the slot has just been preempted. Figure E-2 lists the significance of each code.

If no assignment is present in a given slot, the special codes are checked. Except for the case of a slot out of the demand assignment pool, if the code is non-zero, a display must be generated for that slot. For the case of a pre-

emption, the background is colored red and "PREEMPT" is written in the middle of the block in flashing letters. The remainder of the codes generate either a "CONTROL", a "REQUEST" or a "REQ/REP" ASCII label in yellow characters on a black background.

If the channel is out of the demand assignment pool and this is the first frame since its removal, the slot is blanked out. Further, a red asterisk is placed next to the channel number to provide positive confirmation that the frequency channel has been removed from the demand assignment pool. If the channel is in the pool but previously was removed, the asterisk is erased.

When all of the slots have been processed, the cursor is hidden off the screen so that its presence will not be distracting to the viewer. Once all of these operations have been done, the program proceeds to see if the operator is signalling.

5. Request Queue Display

A flowchart of this function is shown in Fig. 4-12. The following discussion provides more detail.

a. Initialization

When the request queue display is first selected, the previous display is erased, and the heading information is placed at the top of this display. The heading consists of a title, a series of column labels, and a line dividing the column labels from the column entries.

b. Request queue contents display

On each frame for which this display has been selected, the first 20 requests of the current Multiple Access Controller queue are displayed. If more than 20 requests are in the queue, they are not displayed as there is neither space for them on the Intecolor screen nor capacity reserved for such a number in the data stream sent from the MAC to the Intecolor terminal. In general, the queue length does not exceed a half dozen entries; therefore, except in cases of unusual call rates and consequent congestion, all queued requests will be shown on the screen.

Each queue entry consists of four bytes. Unlike the case of the assignment list, this format represents a special condensation of the MAC queue entry structure for the purpose of transmitting the data efficiently to the System Operator's Console. Figure C-5 details the format of a condensed request queue entry.

The queue is transmitted to the Intecolor terminal in sorted form. Requests are ordered by precedence with the lower numbered precedences first. The requests are also displayed in this order.

The first request field decoded is the precedence (1-5). The ASCII code for the number is sent to the CRT screen software. The net address and calling

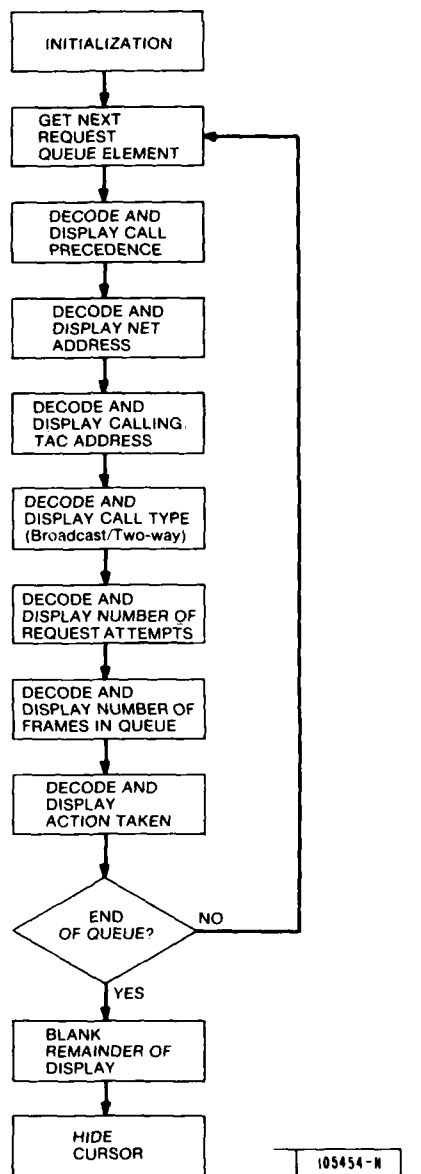


Fig. 4-12. Request queue display flowchart.

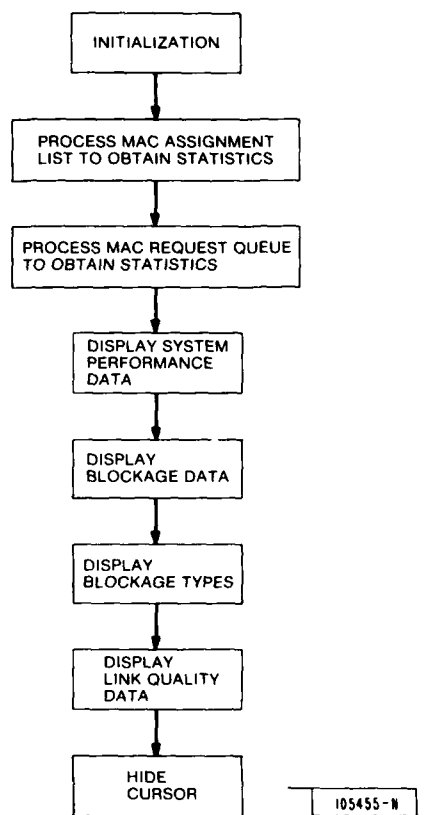


Fig. 4-13. System statistics display flowchart.

TAC addresses are handled in a similar way. The numbers are converted to BCD and thence to ASCII and displayed, this time with three digits rather than a single one as in the case of precedence. The addresses are assumed to be in the correct range. No validity checking is done.

Next the broadcast/two-way bit is unpacked. The bit results in a ASCII message corresponding to its setting. A zero denotes a broadcast call and a one signifies a two-way call.

The number of previous request attempts and the number of frames in queue are unpacked and displayed in a straightforward manner. It should be noted that the number of frames in queue will be zero on the first frame in which a request appears.

The action taken by the MAC on the request is obtained from the request queue entry as a four-bit code. The code is used to table lookup the action taken. The 18-character ASCII message corresponding to the action is then displayed.

Each frame, the entire queue is shown. Because the length of the queue may grow and shrink dynamically, lines after the current end of the queue must be blanked in order to erase any queue entries left over from the last frame. Rather than maintaining a count of lines to be blanked, the program blanks all lines past the current end of the queue.

Once the entire queue has been displayed, the cursor is hidden and the program proceeds to check to see whether the operator is signalling.

6. System Statistics Display

A flowchart of this segment of the display program structure is shown in Fig. 4-13. The means of computing the statistics is explained below.

a. Initialization

Initializing this display consists of erasing the old display, generating the display heading, drawing the boxes in which the data will be entered, and entering labels. In the title, the number of slots per frame is indicated. Therefore, the program must access the flag which tells whether the five- or the nine-slot frame structure is in use. The corresponding number is then included in the display heading.

b. Computation of System Statistics

The data are divided into four categories for the purposes of display. The categories are system performance, blockage, blockage types, and link quality. The data presented give an overview of the current state of the system by processing the data from the MAC assignment list and request queue. All data displayed are for the current frame only.

"System performance" data displayed consist of the percentage of satellite transponder capacity in use, the number and kinds of calls in progress and the

average wait time for response to a request. The first quantities are obtained by processing the MAC assignment list. The number of slots currently in use is the simple sum of the number of slots assigned to user data circuits and the number used for system overhead. That number is divided by the total number of slots available, which is 45 for the 5-slot frame and 81 for the 9-slot frame. The number of calls in service is a simple count obtained from cycling through the MAC assignment list. As the assignment list includes the called TAC address, which is zero for an all-member call, the Intecolor terminal is able to differentiate between point-to-point and all-member net calls. The total point-to-point and all-member counts are stored separately and displayed in the system performance box after conversion to BCD and ASCII. The average wait time for a request is obtained from operations on the request queue. The number of entries in the request queue (that have just received final action) is first obtained. A sum of those entries' times (in seconds) waiting in queue and retransmitting is then done. Dividing this sum by the number of queue entries and adding 6 s gives the average number of seconds wait time for a request.

The "blockage" data consist of the number of calls preempted this frame as well as of the number of requests in queue and their disposition. The number of calls preempted is the sum of the number of slots showing a special preemption code in the low-order byte of the frame of assignment while all other entries in that assignment list entry were zero. The request queue processing consists first of counting action-taken codes for all queue entries. All requests indicated as having received an assignment are summed. Those still waiting in queue are correspondingly summed. Those with other codes encountered some form of blockage and were not assigned; they are also summed. The results of this processing, converted to ASCII, are then displayed.

More detail is provided for the calls that were blocked. This detail is obtained by individually summing all of the codes from the request queue indicating each of the following types of blockage: terminal busy, satellite busy, terminal off, insufficient link, net inactive, net already active, and invalid request.

The "link quality" segment of the display provides information on the quality of the links of currently assigned circuits and those unsuccessfully requesting circuit assignments. The first line shows the number of calls blocked this frame for insufficient link conditions. This count is the sum, which also appears in the "blockage type" box, of the requests which were denied due to adverse link quality for one or more of the request's parties. The remainder of this segment of the display summarizes the link qualities for the assigned calls. The data are obtained from the assignment list. Running summations of the number of calls at the different combinations of burst and code rate are maintained.

As the same data are shown each frame for this display, there are no erasure considerations. Leading zeroes are displayed for numerical counts, thus simpli-

fyng the updating of the display.

Once all of the data have been updated for the current frame, the cursor is hidden and the program proceeds to see if the operator is signalling.

7. Detailed Slot Data Display

Figure 4-14 is a flowchart of the logic used in implementing the detailed slot data display function. Further discussion follows.

a. Initialization

At initialization time for this display, the previous display is erased and the operator is requested to enter the channel and slot number for which detailed data are to be presented. The terminal blanks the screen and issues the prompting message: "CHANNEL= ". The operator must then enter the desired channel number. The value is checked for validity. If it is not the correct range (1-9), a message appears and the number must be entered again. Once an acceptable channel number has been obtained, the terminal prompts with "SLOT= ". The operator must then input an acceptable slot number. Once the slot number is obtained, the display of the slot contents begins.

Next, the heading, the boxes in which the data are entered and the labels within the boxes are generated. The format of these is similar to that for the system statistics display. Also, the heading indicates the channel and slot number of the slot to be examined. These quantities are saved by the program in RAM for the purpose of the labelling and for the accessing of the appropriate part of the MAC assignment list.

b. Display of the Slot Contents

The appropriate entry in the MAC assignment list is accessed. Locating the entry is done by a series of two multiplications. Normally, divisions and multiplications are avoided in order to save time. Here, however, the remainder of the processing and display generation is relatively fast; therefore, the expenditure of time will not cause the Intecolor terminal to miss a system status data burst from the MAC.

Once the assignment list entry has been found, the net address field is examined. If zero, no assignment is present. In that case, the low-order byte of the frame of assignment field is checked for the presence of a special code. If that is also zero, the program lists the slot as available in the "type of call" box and proceeds to erase all other entries in this and the other boxes.

If the low-order byte of the frame of assignment field shows a non-zero code, the appropriate message corresponding to that code is displayed. The remainder of the "type of call" box and all of the remaining three boxes are then completely erased. If the code denotes a preemption, the message is written in red.

In order to avoid erasing entries repeatedly, the status of the slot on the previous frame is saved. Before proceeding to erase the boxes, this

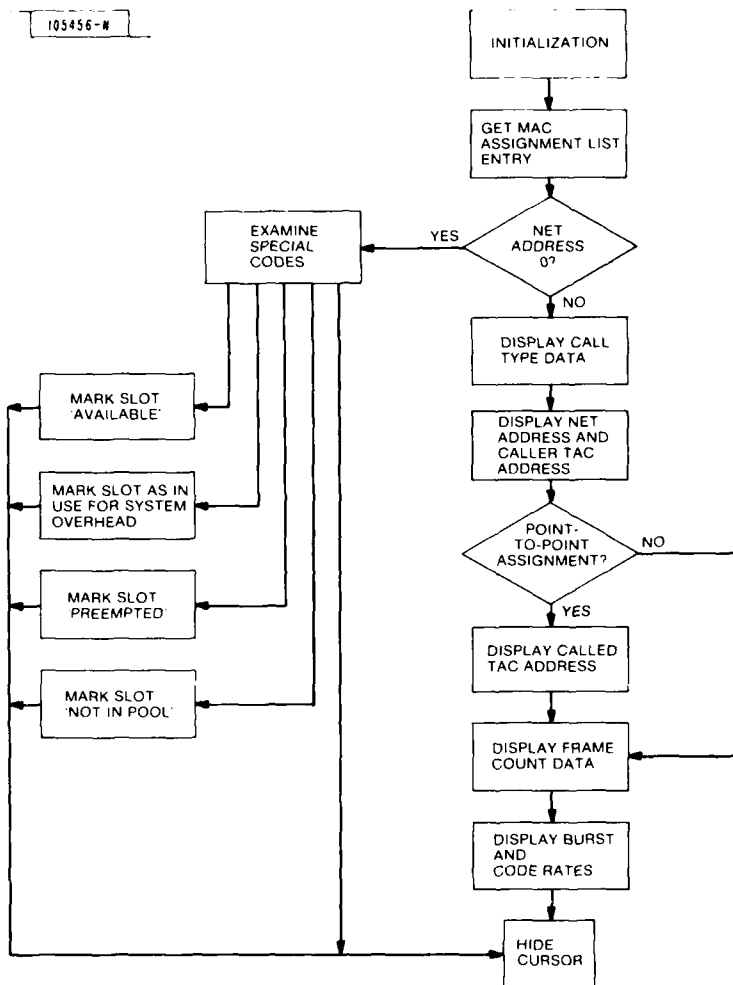


Fig. 4-14. Detailed slot data display flowchart.

old status is checked. If it agrees with the current one, the erasure has already been done and is not repeated. Further, because of the amount of erasure required, the erasure is done by blanking the entire screen and restoring the heading, boxes and the single line containing the slot status. It is therefore important that the erasure only be done once; otherwise, the display would flash on and off at each frame.

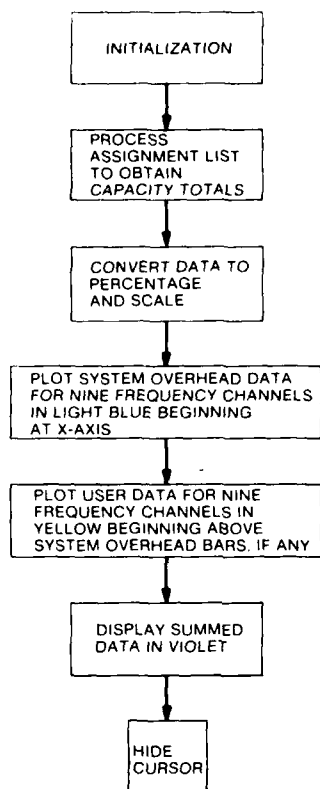
If the net address field is found to be non-zero, a call is occupying this slot. If so, the first line in the "type of call" box is immediately erased in the event that a message was already present on that line.

The "type of call" box is then updated to display the call precedence, the number of slots used for the circuit, whether the call is a broadcast or a two-way call, whether an all-member or point-to-point net is using the circuit, and whether (for the point-to-point case) it is a conference call. All of these data are obtained from the MAC assignment list.

The call precedence is directly translatable to a display format by a simple binary-to-ASCII conversion. The number of contiguous slots for the circuit in the MAC assignment list is converted to ASCII and then displayed. The remaining quantities are displayed as ASCII messages for the convenience of the operator.

The broadcast/two-way indicator is a single bit: If it is zero, "broadcast" is written; while if one, "two way" is written. Care is taken to pad the "two-way" message to the length of the "broadcast" one so that when the former replaces the latter the extra characters are blanked out. The situation is similar for the all-member and the point-to-point net indicator. The distinction is made by testing the value of the called TAC address in the assignment list. If zero, the call is an all-member one. Again, care is taken to pad the ASCII messages to the same length to avoid leaving part of the old message on the CRT screen when a change of assignment occurs in this slot. If the net is a point-to-point one, the call may be a conference call. In that event, a bit will be set in the assignment list entry. If this bit is set and the call is a point-to-point one, an additional line will appear on the display - "conference call". If the call is either an all-member one or not a conference call, the line is erased.

The upper right box of this display (see Fig. 4-7) records the net and TAC addresses of the parties participating in the call. All addresses can potentially occupy three digits. Therefore, for simplicity in displaying them, leading zeroes are displayed. They are obtained from the MAC assignment list, converted to BCD, and thence to ASCII for display purposes. The third line in this box shows the called ("callee") TAC address, a quantity which is not meaningful for all-member net assignments. If the called TAC field is zero in the MAC assignment list, this line is erased on the screen, and only the net and caller TAC addresses are shown.



105457-N

Fig. 4-15. Satellite capacity utilization bar graph flowchart.

Frame counts are displayed in the lower left box. Remaining, current, and initial counts are displayed. The "remaining" counts are obtained by subtracting from the call duration the difference between the current frame of day and the frame of assignment. The current frame number is passed to the Intecolor terminal in the data stream from the MAC. The other quantities, the frame of assignment and the call duration, are found in the assignment list. All of these three quantities are displayed with five digits so that the maximum (16-bit) frame count allowable by the MAC may be represented.

The remaining box in the display shows the burst and code rates for the assignment. These quantities are encoded in the assignment list, and the uncoded equivalent is presented in ASCII in the box. The codes and their meanings are as follows:

burst rate		code rate	
code	meaning	code	meaning
0	-----	0	1/2
1	9600 sps	1	2/3
2	19200 sps	2	3/4
3	32000 sps	3	4/5

Once the display is complete, the cursor is hidden and the program checks to see if the operator is signalling.

8. Satellite Capacity Utilization Display

The flowchart for the satellite capacity utilization bar graph program segment is shown in Fig. 4-15. A discussion of that program follows.

a. Initialization

During the first frame of this display, the previous display is erased, the background is changed to dark blue, and the axes, tic marks, and key information are entered on the display.

b. Bar Graph Generation

The data for this bar graph are collected each cycle immediately after the burst of data is received from the MAC. A subroutine processes the MAC assignment list and stores three types of data in RAM for the display routine. For each channel, the subroutine records the number of slots occupied by both user data and system overhead data. It then sums those quantities and stores the total number of occupied slots in RAM for the display routine. This subroutine collects the data in raw form. The display routine then further processes these data so that they are both converted to a percentage and scaled appropriately for display generation.

This bar graph consists of ten bars plotted upwards for the x-axis. The leftmost nine bars represent the percentage of the given frequency channel

currently in use. They are graphed in two colors, light blue and yellow. Light blue represents system overhead (signalling) data, while yellow indicates user data. The tenth bar represents total data and is graphed in violet. See Fig. 4-8 for a sample display.

This graph is generated by first plotting the system overhead data, by then plotting the user data and finally by adding the total data. Due to the narrowness of the bars produced by the hardware implementation of the terminal bar graph option and due to color change restrictions which limit the resolution to the size of a small ASCII character, the hardware bar graph option was not used. Instead, the character generation mode is used, and pairs of blanks are used to form the bars. The bars are generated by a series of 25 pairs of blanks as the pairs provide for an appropriate bar width. For the area of the screen used, this subdivision represents the maximum resolution possible in the y direction for updating in a real-time mode. The real-time aspect of the graph generation adds a constraint: The bars must be capable of dynamically growing and shrinking. Thus, the program must be able to erase parts of bars dynamically. Due to the means by which the color generation is done, when one part of a character-sized block (which may be sub-addressed by the bar graph mode) has its color changed due to an erasure, the remainder of that block changes color also, thus limiting the resolution to that of an ASCII character. Therefore, the character mode was used so that colors may be changed in a straightforward manner.

Once the bar color is selected (light blue, yellow, or violet), the y coordinate, ymax, of the top of the bar is calculated. This coordinate is obtained from RAM. These counts are then multiplied by the appropriate factor to convert the data simultaneously to percent and to the number of blanks necessary to display in order to reach the desired ymax value. Table lookup is done in place of multiplication. For the case of system overhead data (which is plotted first), ymax is saved so that the user data bar will begin at the top of the system overhead bar.

Next, the 25 pairs of blanks are displayed, beginning at either the x-axis or the previous ymax. Erasure is done by changing the color of the blank to the color of the background (dark blue) when the ymax value is reached. Bars are generated once for the system overhead data and once for the user data for each of the nine frequency channels.

The last bar graphed is the one reflecting the total usage of satellite capacity. As for the others, the count has been stored in RAM and need only be converted. The bar is then drawn in a similar manner to the others with the color changing to the background color at the appropriate point.

Finally, the cursor is hidden, and the program jumps to the location at which operator signalling is polled.

9. Call Precedence Distribution Display

Figure 4-16 depicts the flowchart for this display function.

a. Initialization

At initialization time for this bar graph, the previous display is erased, the background color is set to black and the axes, tic marks and labels are generated.

b. Precedence Distribution Display

This bar graph shows the current distribution of assigned slots by precedence. The bars are color coded for each of the five call precedences in the same manner that colors are used in the frame structure display. Like all the other displays, this one is updated each frame; the bars grow and shrink as the distribution of precedences changes.

The number of slots at each call precedence is first computed. In order to accomplish this, the program cycles through the assignment list, examining the call precedence field only. If no call is occupying a slot or if the slot is used for system overhead, the precedence bits will be zero. Therefore, the program simply checks this field and totals the number of assignment list entries containing each.

The bar graph is generated next. Its means of generation and scaling are the same as for the satellite capacity utilization bar graph. The principal differences are that the individual bars are in different colors and that no scheme of changing the color part way up the bar is required. The erasure scheme remains the same; the color is changed to the background color (in this case, black) when the top of the bar is reached. Twenty-five pairs of blanks are displayed for each bar as in the previous case.

When done, as in all displays, the cursor is hidden, and the program branches to the location at which operator signalling is polled.

10. Polling the Keyboard for Operator Signalling

By the use of a CRT operating system utility, the display program is able to poll the System Operator's Console keyboard to see if the operator is currently striking a key. If s/he is not, the program proceeds back to receive the next burst of data from the MAC minicomputer.

If the operator is striking a key, the program reads the value and checks to see if it is a question mark, which is the designated "attention key". The question mark was selected because it requires depressing the shift key and an additional key simultaneously and is therefore not likely to be hit accidentally. Further, the shift key and the "?" key are next to each other. Therefore, without awkwardness, the operator may hold down this key until the terminal responds.

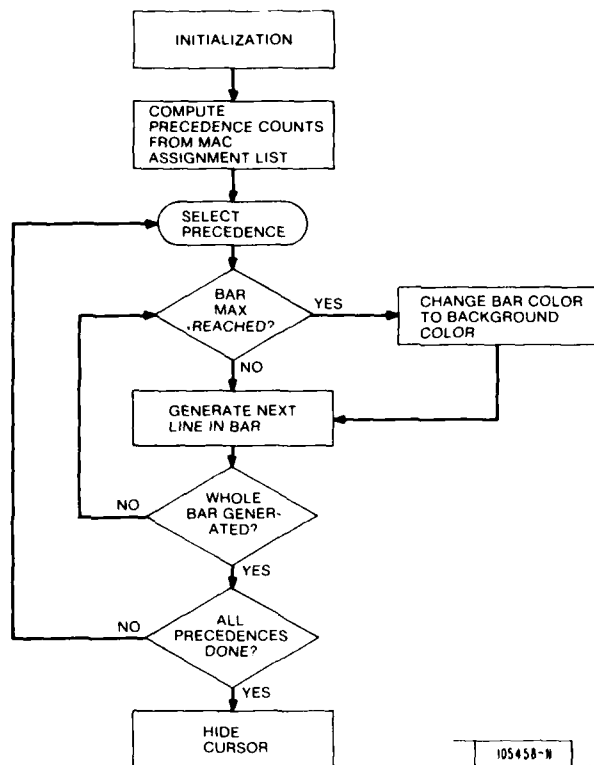


Fig. 4-16. Call precedence distribution bar graph flowchart.

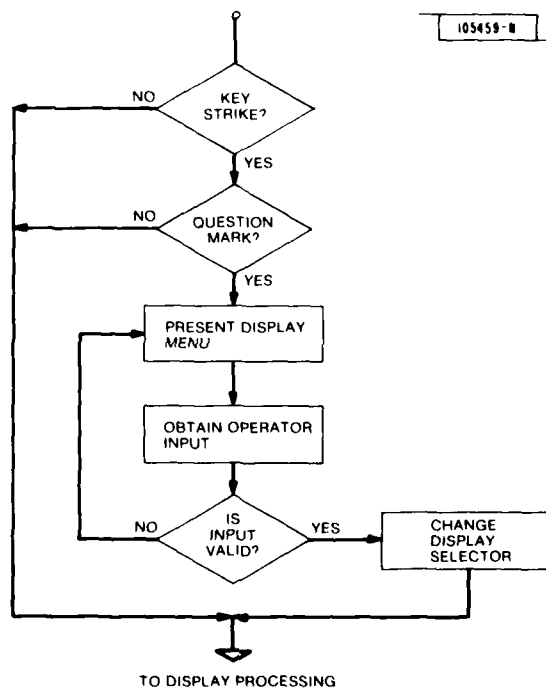


Fig. 4-17. Keyboard polling flowchart.

Because the polling of the key strike is done only once per frame, the operator must depress the key continuously until the program reaches this section of the code. In actual operation, this usually occurs within a second and is therefore not a great inconvenience (although an interrupt-driven event would provide faster service).

If a key was struck and it is not a question mark, the program assumes the key was struck accidentally and returns to receive input from the MAC. If the character is a question mark, the display menu is presented to the operator (see Fig. 4-2). The operator should then enter a number from 1 to 7. The values 1-6 will result in the generation of the display listed next to that number; the value 7 will cause the terminal to enter the command mode.

The terminal checks the operator input. If valid, the program proceeds immediately to the generation of the requested display or to the command mode. Receiving of the data burst from the MAC is deferred at this point so as to give faster service to the operator. On the next cycle, the program will return to the data receiving mode. If the operator input is not within the acceptable range (1-7), the operator must make another selection.

The flow of this section of the program is shown in diagrammatic form in Fig. 4-17.

11. Command Mode

When the command mode is entered, the terminal expects input from the operator. The following commands are recognized:

<u>Command</u>	<u>Shortest Abbreviation Permitted</u>
DELETE n	Dn
ADD n	An
PREEMPT CH n SL m	Pnm
ESCAPE	E

where n is a valid channel number (1-9) and m is a valid slot number (0-4 or 0-8, depending on the frame structure is use). The terminal accepts a line of input from the operator. When a carriage return has been received, the program parses the input data stream, processing a maximum of 18 characters.

Command recognition is accomplished through examination of the first character in the input buffer. If it is a "D", "A", "P", or an "E", the command is recognized as valid. Otherwise, the input is rejected, and the operator must enter another command. Once a command has been recognized, the program identifies the first two numbers in the input buffer. These values are used as the channel and slot numbers, respectively. If no digits are found or if they are out of the acceptable range, the operator is prompted again and required to reissue the command.

In addition to checking the ranges of the input, validity checking of the command is performed. The Operator's Console checks the inputs to ascertain

whether the action requested is permissible. No commands are sent to the MAC unless they meet certain requirements. The requirements differ depending on which command was issued.

For the addition and deletion of channels, the assignment list is persued. When attempting to delete a channel, the program checks to be sure that the channel is in fact in the demand assignment pool. This verification is done by checking that the special assignment list code indicating a channel's removal from the pool is not present in any of the slots in that channel. Next, the program checks the special codes to ensure that no system overhead slots are in the specified channel. The deletion of such a channel would be a disastrous event for the MAC and is not permitted. If any of these above conditions are not met, the operator command is rejected (with an appropriate message to the operator).

In the case of channel addition, the special codes in the assignment list are checked to ensure that the channel is indeed out of the pool. For preemption, the net address field is checked. If zero, indicating that no call is occupying the slot, the command is rejected. If non-zero, the net address is displayed on the screen before sending the command to the MAC minicomputer.

Once the command has been received, parsed, and checked for validity, the ten-byte command output buffer for the MAC is prepared. This buffer consists of a one-byte command code followed by the channel number and (for the case of a preemption command) the slot number and net address. Figure C-6 contains details of the system operator command data format for the MAC. Both a header and a trailer byte are used in transmitting the data to the MAC. These bytes assure the MAC that it has received the entire command burst from the Intecolor terminal before proceeding to process it. This is particularly important as the MAC does not do extensive validity checking on operator commands; the MAC relies on the System Operator's Console for this function.

There are a number of timing considerations involved in sending a command burst to the MAC. Because the MAC is using the same DMA controller to control both input from and output to the Intecolor terminal, the MAC is not capable of receiving and transmitting Operator's Console data at the same time. As a consequence, the terminal must not send out a command burst to the MAC when the MAC is transmitting data to it. To ensure that this condition is met, whenever a command burst is to be sent to the MAC, the terminal first enters a wait loop to receive the next data burst from the MAC. Once that burst has been received, the terminal can be certain that the MAC is not transmitting data. The terminal then delays for approximately 30 ms to allow the Varian minicomputer sufficient time to change the data direction of the DMA controller. The terminal then transmits the ten-byte burst containing the command.

It should be noted that the sending of the command burst is not a regular occurrence. The burst is only sent when the operator has entered a command,

and it is therefore a relatively infrequent event. The MAC examines its input buffer from the Intecolor terminal each frame, but there is seldom data to process.

Once the command has been sent to the MAC in the Varian minicomputer, the display program returns to the frame structure display so that the operator may monitor the effect of the command that was issued.

The structure of the programming of the command function is illustrated in Fig. 4-18.

F. Display Generation on the Intecolor 8051 Color Terminal

The Intecolor 8051 terminal is a RAM-refreshed CRT. Two bytes of status and display data are maintained for each screen location. Display generation is thus done by modifying the contents of the RAM refresh buffer.

If the manufacturer-provided BASIC interpreter is used, the explicit updating of the RAM refresh buffer is done by the interpreter. When programming in Intel 8080 assembly language, the user must interact much more directly with the display hardware. There are, however, operating system utilities which are accessible to the user program.

These operating system utilities allow the user to encode the display data into ASCII codes from 0-31 and 240-255. The screen software will then interpret these codes and perform the required operations on the RAM refresh buffer, thus freeing the user from the complex bookkeeping required in order to accurately update the refresh buffer. The low-numbered ASCII codes allow the user to change colors, position the cursor, change character size, enter Intecolor processing subsystems, etc. The high-numbered codes are provided for plotting points, vectors, and bars.

The TACS display program employs these utilities for all of its display generation. The program sends either a single character or a string to the screen software; it is then interpreted and refresh RAM is updated accordingly. As software processing is necessary in order to affect the CRT display, the response time of the terminal is not extraordinarily fast. It is, however, sufficiently rapid to permit the updating of system status displays in real time.

G. Input/Output Handling on the Intecolor 8051 Terminal

The input/output handling for the 8080 CPU contained in the Intecolor 8051 terminal is accomplished by a TMS 5501 Controller. The TMS 5501 provides the microprocessor system with an asynchronous communications interface, data I/O buffers, interrupt control logic, and interval timers. Commands to the chip allow the 8080 CPU to read the RS-232 receiver buffer, the input port, the interrupt address, TMS 5501 status, to issue discrete commands, and to load the baud rate register, the transmitter buffer, the output port, the mask register, and an interval timer.

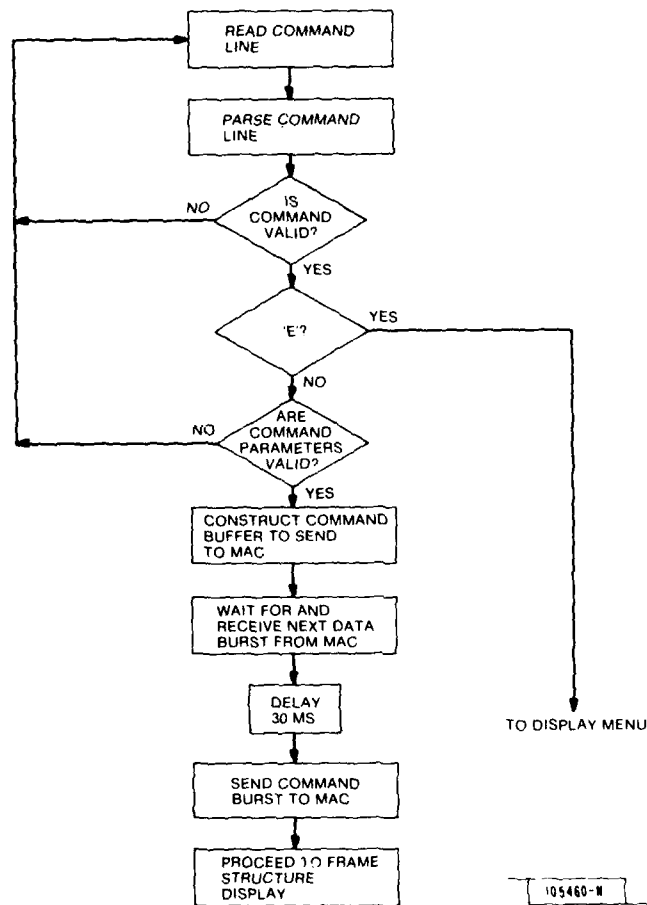


Fig. 4-18. Command mode flowchart.

For the TACS display program, RS-232 input/output handling is the principal area of interest within the controller. By use of Intel 8080 input and output instructions, a user program may communicate with the TMS 5501 controller. Specifically, the TMS 5501 decodes the port number specified by the input or output instruction and correspondingly provides or accepts the appropriate data. In this way, a program may handle its RS-232 input.

H. Program Development Information

The source code for the TACS System Operator's Console consists of approximately 4500 lines of Intel 8080 assembly language code. The program occupies 12K bytes of RAM in the Intecolor terminal. Due to the size of the program, program development was done on the Laboratory central computing facility, an IBM 370/168. A cross assembler purchased from the Intel Corporation was used for this purpose. The Intel 8080 cross assembler output consists of an ASCII representation of the machine code. That machine code is then downloaded into the Intecolor terminal via its RS-232 port. A special program in the Intecolor terminal reads data from the serial port, converts it to actual machine language and stores it in RAM. From there, the machine code is stored on flexible diskette by using the floppy disk operating system provided by the manufacturer of the Intecolor terminal.

V. SCENARIO GENERATOR

A. Motivation for Automated Scenario Generation

An essential part of bringing TACS on the air is the creation of a "scenario" or "operating environment." A given scenario is characterized by the set of subscriber units that can access the system and the manner in which these units' I/O ports are organized into communications nets. The configurations of the subscriber units may vary: They may be linked to other units on the same platform or may be the only unit on the platform; they may possess full- or half-duplex transceivers; they may access the primary or alternate control and request/report subslots; etc. The characteristics of the communications nets in a scenario may also vary: They may be all-member or point-to-point nets; they may handle record or voice data; they may or may not have a net controller; etc.

The operation of TACS requires that every subscriber unit have information (in its memory) describing its place in the current scenario. This information includes the TAC address assigned to the unit, the control community (control, ranging, and request/report subslots) it must access, the addresses and characteristics of any nets with which its I/O ports may be affiliated, and the TAC addresses of the other members of these nets. The current implementation of TACS does not provide any automated means for a TAC unit's acquisition of this information; pre-arranged scenarios are assumed. (In a future system, however, it may be possible to transmit much of this information via specially designated satellite circuits as part of system startup or reconfiguration.)

The MAC must have access to information on all aspects of the current TACS scenario. Such information is contained in two of the elements of the MAC's data base: the TAC list and the net list. Copies of these two lists also provide the Call Simulator with scenario information. Appendix E, "Multiple Access Controller Data Base," presents detailed descriptions of the TAC and net lists.

TACS Central Control Facility testing has proceeded in three phases, each phase requiring the construction of a number of TACS scenarios. In the first phase, the proper operation of the software components of the MAC, Call Simulator, and System Operator's Console was verified. The verification required highly nonuniform scenarios designed to exercise as many of the software branches as possible. The second phase consisted of demand assignment performance tests, each test designed to emphasize a single type of TACS blockage (e.g., satellite blockage, I/O port blockage). These tests were structured so that their results could be easily matched against analytical predictions, thereby providing another level of software verification. Uniform scenarios were required. The third series of tests also made use of uniform scenarios. This series presented views of overall system performance, stressing the efficiency gains obtained via time division multiplexing and demand assignment.

The performance testing results are described in Taylor and Bernstein.⁴

The requirements of the testing program, and the difficulties encountered in manually assembling TAC and net lists indicated the need for some automated form of scenario generation. A scenario generating program would not only simplify TACS verification and performance testing, but would allow the creation of (pseudo-) realistic scenarios that could be used with the real TAC units for demonstrations of typical system operations. In addition, a Scenario Generator would greatly simplify the reconfigurations that would at times be required in a fully operational system.

B. Description of Scenario Generator

The TACS Scenario Generator runs on the same Varian 620/L-100 minicomputer as does the MAC. It is a non-real-time routine, accepting user input via the Tektronix or NCR terminals. Its function is to assemble, according to such user input, a TAC list and net list that can be used for MAC and Call Simulator operations. The Scenario Generator ensures TAC and net list consistency; the data entered by the user in building these lists must be consistent with all previously entered data. Improper or inconsistent inputs generate error messages. The TAC and net lists generated by the program are always properly cross-referenced, and represent a physically realizable scenario. For example, only as much capacity is reserved at a TAC's transceiver for all-member nets as the TAC has available. Any attempt to reserve capacity greater than the transceiver's limit results in an error message. Some elements of the TAC and net lists, not being a part of the scenario, are altered dynamically during MAC and Call Simulator operations (e.g., the number of sequential status report misses in the TAC list, and the on/off the air bits in the net list). The Scenario Generator properly initializes the data in these variable fields. Scenarios generated are stored on magnetic tape, and can be easily transferred to floppy disk for use by the Call Simulator. Hard-copy records of the scenarios can be output.

As indicated in the flowchart in Fig. 5-1, there are three phases to the scenario generation process: initialization of the TAC list, definition of the characteristics of communications nets, and building the membership of these nets. Reference to Appendix E will aid in understanding the following discussion of this process.

TAC list initialization involves determining the number of TACs in the system (63, 127, 255, or 511), and formatting the corresponding number of TAC list blocks (see Fig. E-3). All blocks initially have an "off" status, and have no net affiliations. TAC addresses 1 through 9 are reserved for the TACs at the central control site. The Scenario Generator automatically places the nine corresponding TAC list blocks in "on" status, flags all central control site transceivers as full duplex, indicates membership in both the primary and alternate control communities, and prompts the user for the central control

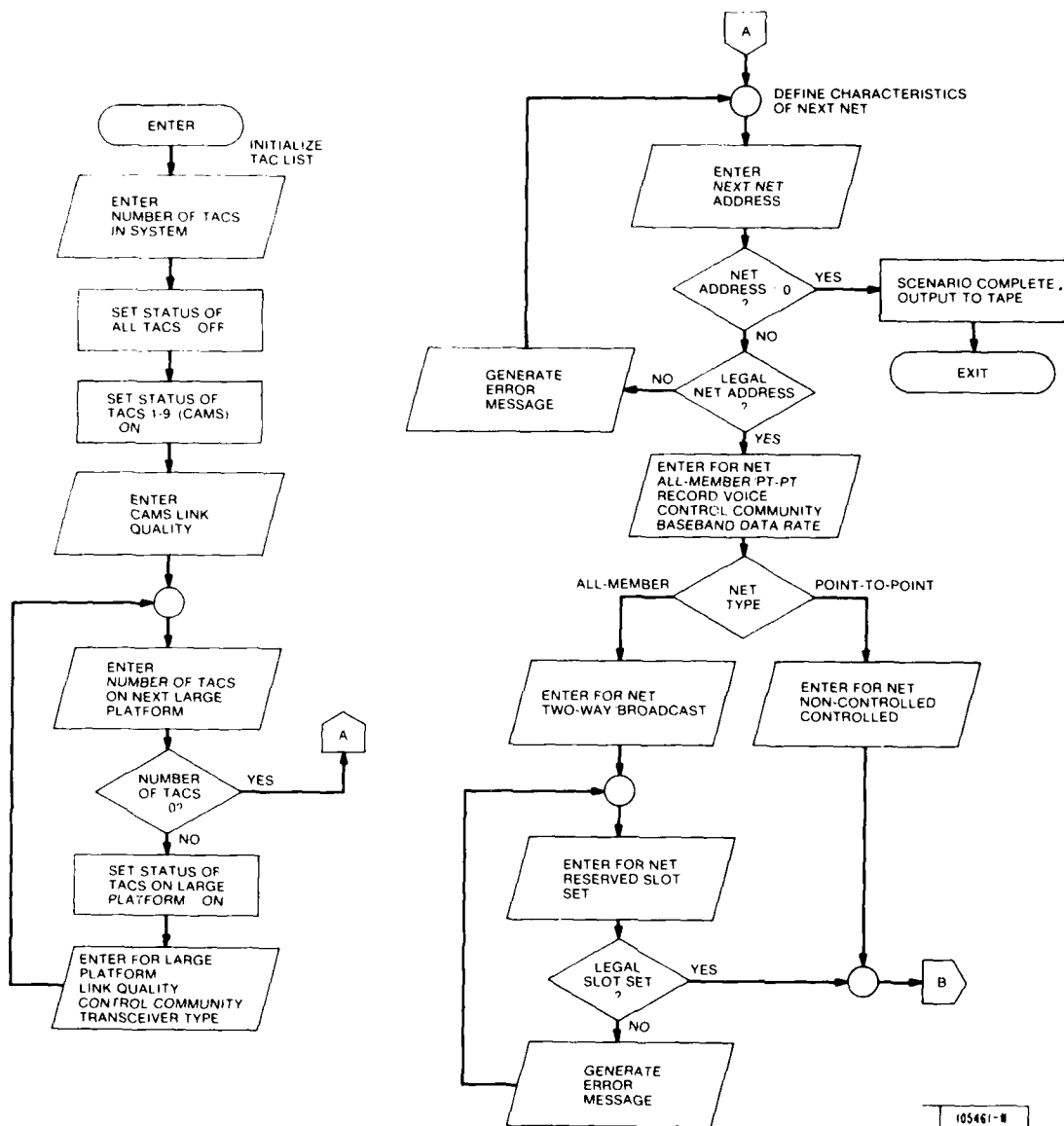
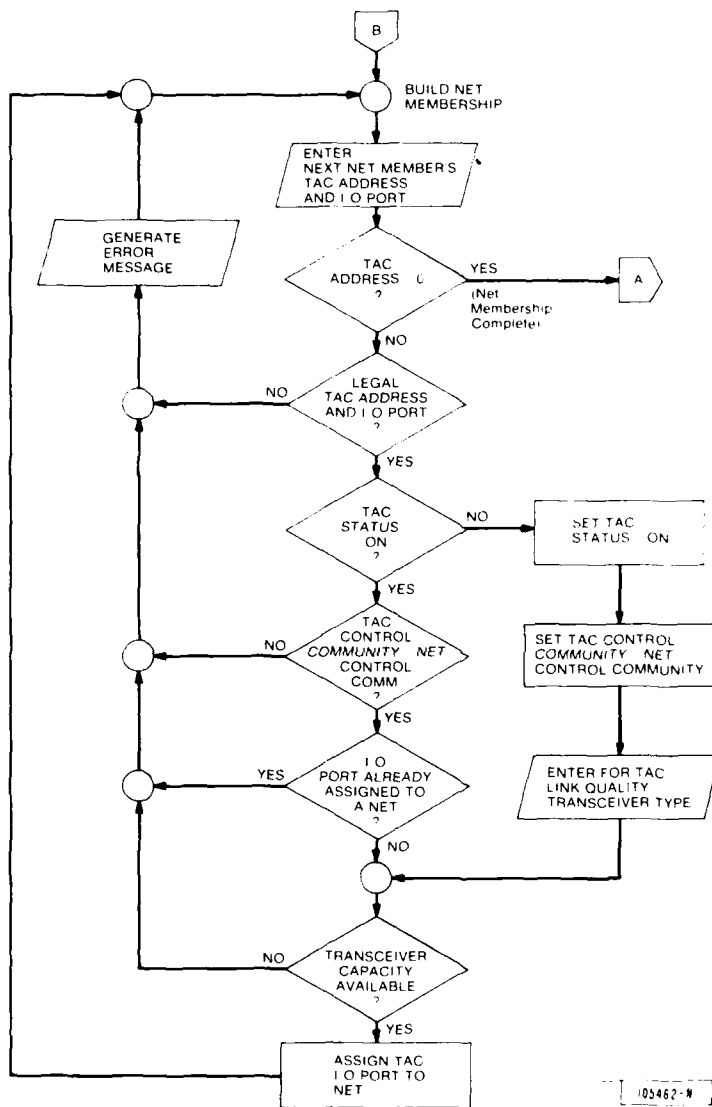


Fig. 5-1. Scenario generator flowchart.



105462-N

Fig. 5-1. Continued.

site link quality (the uplink and downlink signal-to-noise ratio and percent RFI to be inserted in the nine TAC list blocks). The user of the Scenario Generator next defines any "large" platforms (platforms with multiple TACs and associated transceivers) that will be in the scenario. The number of TACs on the platform and the platform's link quality, transceiver type (full- or half-duplex), and control community (primary, alternate, or both) must be supplied by the user.

The definition of the characteristics of a communications net involves supplying much of the information resident in the first two words of a net list block (see Fig. E-4). The net address is checked for validity: It must be a previously unused address that is within the legal range of addresses (1-127). The slot set to be reserved (at the net members' transceivers) for an all-member net is also checked: The nine-slot TDMA frame structure only handles slots numbered from 0 through 8.

Once a net's characteristics have been defined, the user must build the net's membership by keying in TAC addresses and I/O port numbers. A number of tests are made before one of these TAC I/O ports is added as a net member. These tests constitute the most complex part of the Scenario Generator. If any of the tests indicate that the I/O port cannot be included in the net, an error message is generated giving the user the reason for failure. If all tests are passed, the new net member is indicated in the appropriate TAC and net list fields, and the user is prompted for another net member.

The first test performed on a prospective net member concerns the validity of the TAC address and I/O port number specified. I/O port numbers must be between 0 and 3; TAC addresses must be between 1 and the number of TACs in the system. In addition, the net's membership cannot already contain an I/O port with the same TAC address. (TACS does not allow a single TAC unit to have two or more I/O ports affiliated with the same net.) If the TAC list block associated with the TAC address specified is in "off" status, its status is changed to "on," it is placed in the same control community as that of the net currently being defined, and the user is prompted for the transceiver type and link quality. If the TAC's status is already indicated as "on," its (previously defined) control community must be the same as the net's control community. A test must also be made to verify that the I/O port specified for such a TAC is not already affiliated with another net.

Prospective net members' transceivers must be checked to verify that they have the capacity required to support the net. In the case of all-member nets, the capacity needed to communicate in the net's reserved slot set must not have been previously reserved at the transceiver by other nets. Although point-to-point nets do not reserve transceiver capacity, for such nets the Scenario Generator verifies the transceiver's ability to support a two-way circuit in at least one TDMA slot (after the capacity previously reserved by all-member nets has been accounted for). The analysis of transceiver capacity uses techniques similar to those employed by the point-to-point net branch of the MAC's assignment algorithm (see Section II.G.3 and Appendix G).

ACKNOWLEDGEMENTS

The TACS project involved the significant efforts of a large number of MIT Lincoln Laboratory people. Of particular assistance to the Central Control Facility work described in this report were: Chris Moulton and Mark Saklad for work on the MAC software, Willie Brown for Call Simulator software support, Dale McNeill for guidance on the Varian minicomputer operating system, and Arthur Levasseur and Bill Stotz for Varian hardware maintenance. Acknowledgements are also due Linda Jeffrey, Anne Mead, and Barbara DiPalma for typing the report.

REFERENCES

1. J. D. Bridwell and I. Richer, "A Preliminary Design of a TDMA System for FLEETSAT," Technical Note 1975-5, Lincoln Laboratory, M.I.T. (12 March 1975), DDC AD-A007823/8.
2. J. D. Bridwell, "A Terminal Access Control System for FLEETSAT," Technical Note 1976-29, Lincoln Laboratory, M.I.T. (22 November 1976), DDC AD-A034813/6.
3. S. L. Bernstein et al., "Variable-rate Viterbi Decoding in the Presence of RFI," Vol. 3, IEEE National Telecommunications Conference Record (December 1977).
4. L. E. Taylor and S. L. Bernstein, "TACS - A Demand Assignment System for FLEETSAT," IEEE Trans. Commun. COM-27, (1979), pp. 1484-1496.
5. S. N. Landon et al., "The TACS Subscriber Unit and Digital Satellite Simulator," Technical Report 540, Lincoln Laboratory, M.I.T. (22 April 1980), (to be published).
6. I. Richer and D. A. McNeill, "Summary of the System, Support, and Special-Purpose Software Used by the Sanguine Simulation Facility," Technical Note 1972-31, Lincoln Laboratory, M.I.T. (18 September 1972), DDC AD-753115.
7. D. E. Knuth, The Art of Computer Programming, Vol. 2. Seminumerical Algorithms (Addison-Wesley, Reading, Massachusetts, 1969), pp. 101-121.

GLOSSARY

Terms

All-Member Net	One of the two principal kinds of network configurations managed by TACS. All net members participate (as receivers and/or transmitters) in any net activity. Such nets are always controlled.
Assignment	Dedication of TDMA slots(s) to members of a net for communications purposes.
Assignment Algorithm	MAC algorithm that services requests pending in queue, attempting to make assignments.
Assignment List	MAC data base component providing a detailed description of the usage of every TDMA slot in every channel.
Assignment Message	Message issued in the control burst informing net members of a circuit assignment.
Broadcast Circuit	Circuit in which one party <u>always</u> transmits and all other parties <u>always</u> receive.
Broadcast Net	All-member net constrained to establish <u>only</u> broadcast circuits (with the net controller as the transmitting party).
Call	Usually a synonym for "request"; in some contexts it may imply the same meaning as "circuit."
Call Duration	Specified time requirement for a circuit.
Call Initiator	Net member requesting a circuit assignment from the MAC.
Call Precedence	One of five levels of "importance" associated with requests and subsequent assignments.
Call Precedence Threshold	Minimum call precedence required for newly generated requests; it is specified in the control burst.
Call Progress Message	Message issued in the control burst informing a call initiator of the inability of the MAC to issue a circuit assignment.
Call Simulator	PDP-11/03-minicomputer-based simulation of a large community of TAC units.
Central Control Facility	Consists of the following TACS components: MAC, Call Simulator, System Operator's Console, Master TAC.
Channel	One of the nine 25-kHz-wide satellite frequency channels managed by TACS.
Circuit	A TDMA slot or a set of TDMA slots assigned to net members for communications purposes.
Circuit Assignment	See "assignment."
Circuit Request	See "request."
Communications Area Master Station	Communications site with the MAC, and up to nine TAC units (TAC addresses 1-9) with associated transceivers. Such a station would exist in a full-scale system. Sometimes called a central control site.

Conference Call	Circuit involving net members other than the original calling and called parties. Only point-to-point nets may establish conference calls. In some contexts "conference call" may refer to a request for such a circuit.
Control Burst	Data sent in the control subslot performing system control functions.
Control Community	Indicates the set of control, ranging, and request/report subslots accessed by a TAC unit; e.g., a TAC unit belongs to the alternate control community if it accesses the alternate set of subslots.
Control Subslot	Subslot used for the transmission of control burst data to TAC units. All units must continually monitor this subslot.
Controlled Net	Point-to-point net with a party designated as the net controller. Such nets may only support one circuit at a given time.
Data Slot	See "slot."
Debug	Varian minicomputer's operating system.
Dedicated Subslot	Ranging or request/report subslot assigned to a given TAC unit in a given frame. Such subslots regularly cycle through the entire TAC community. Status reports are normally transmitted in dedicated request/report subslots.
Demand Access Controller	Intel-8080-based segment of a TAC unit.
Demand Assignment Controller	See "Demand Access Controller."
Demand Assignment Pool	Set of TDMA slots, in all channels, that are currently available for assignment to nets.
Digital Satellite Simulator	Digital (i.e., baseband) hardware built at Lincoln Laboratory to simulate a real satellite link between the Master TAC and remote TAC.
Elastic Area	Segment of the control burst used in a pooled fashion for messages directed to specific TAC units.
End User	Person or device generating the circuit requests sent to the MAC, and the data transmitted in assigned slots.
Fast Processor	AMD-2901-based segment of a TAC unit.
Frame of Day	Count of consecutive TDMA frames maintained by all units in TACS; it is used for system synchronization functions.
Guard Bands	Overhead (extra time) included as part of every TDM slot (or subslot) to allow for the maximum possible satellite range and clock errors expected of any subscriber unit.
Guard Intervals	See "guard bands."
Inelastic Area	Segment of the control burst that is of a fixed format and is directed to <u>all</u> TAC units every frame.

Initial Range Correction Message	Message issued in the control burst informing a TAC that has just issued a ranging burst of its correct satellite range.
Input Algorithm	MAC algorithm that processes ranging bursts and status reports and does initial processing on requests.
Interactive Circuit	Circuit in which all parties may both transmit and receive.
Interactive Net	All-member net that may establish interactive circuits.
I/O Port	As applied to TACS, refers to the four TAC unit port through which end users generate and receive the data transmitted in assigned slots.
I/O Port Number	Number (0-3) uniquely associated with each I/O port on a given TAC.
Large Platform	Platform with <u>multiple</u> TACs and associated transceivers.
Late Entry Request	Request sent from a newly powered-up member of an all-member net desiring entry into the net's ongoing communications, if any.
Link Calculation	MAC's process of determining the burst and code rates required to establish a circuit, such rates guaranteeing adequate error protection for all parties to the circuit.
Master TAC	Lincoln Laboratory built TAC unit communicating with the MAC via a hard-wired link, and with the remote TAC via the Digital Satellite Simulator.
Message Duration	See "call duration."
Message Precedence	See "call precedence."
Multiple Access Controller	Varian-minicomputer-based TACS central controller.
Net	Communications network organization supported by TACS. A subset of TACS end users comprise a net. Nets are of a number of types and sub-types (all-member or point-to-point, record or voice, etc.).
Net Address	Number (1-127) uniquely associated with each net.
Net Controller	Net member (of an all-member net or a controlled point-to-point net) specially designated to monitor all net transactions and/or enforce the net's protocols when the net is assigned a circuit.
Net List	MAC data base component describing the characteristics and membership of each net.
Operating Environment	See "scenario."
Override Request	Request for a new circuit generated by a net member already actively involved as a receiver in an ongoing all-member net broadcast circuit. Such a request is issued by the net member in order to gain transmit capability.
Platform	A single communications site, containing one or more TAC units and associated transceivers.

Point-to-point Net	One of the two principal kinds of network configurations managed by TACS. Circuits are assigned involving subsets of the net membership, usually only a called and calling party. If the net is non-controlled, more than one such circuit may exist simultaneously.
Polled Subslot	Ranging or request/report subslot assigned to a given TAC unit in a given frame via a polling command. Status reports may be transmitted in polled request/report subslots.
Polling Command	Message issued in the control burst requesting action by a specific TAC unit: Such action is normally a response in a specific (polled) ranging or request/report subslot.
Preemption Message	Message issued in the control burst commanding all TAC units communicating in a given TDMA slot (in a given frequency channel) to terminate such communications.
Preemption Threshold	Employed by the MAC assignment algorithm to specify the currently allowable level of virtual (satellite or transceiver) preemptions that may be used in trying to assign system capacity to a request. A preemption threshold may be iteratively raised up to the level of the request's call precedence.
Primary Request/Report Subslots	Request/report subslots that are located in the same channel as the control subslot.
Random Access Subslot	See "shared subslot."
Range Update Message	Message issued in the control burst informing a TAC that has just issued a status report of its correct satellite range.
Ranging Burst	Data sent in a ranging subslot by a TAC unit initially entering the system. Used by the MAC to issue an initial range correction message.
Ranging Subslots	Subslots used for transmission of ranging burst data to the MAC.
Real TAC	One of the two TAC units built at Lincoln Laboratory (the Master TAC and the remote TAC).
Record Net	Net whose communications (user) data consist entirely of automated record data.
Remote TAC	Lincoln Laboratory built TAC unit communicating with the Master TAC via the Digital Satellite Simulator.
Report	See "status report."
Report Holding Buffers	Call Simulator buffers used to hold the status reports to be sent to the MAC within the next three frames.
Report/Request Subslots	See "request/report subslots."
Request	Data sent from a net member to the MAC, in a request/report subslot, specifying the need for a circuit.

Request Parameters	Parameters supplied by a net member as part of a request, indicating specific characteristics of the request. The parameters are required by the MAC's assignment algorithm.
Request Queue	MAC data base component consisting of an ordered list (by call precedence and frame arrival) of pending circuit requests.
Request/Report Subslots	Subslots used for transmission of requests and status reports to the MAC.
Retransmission Holding Buffers	Call Simulator buffers used to hold requests for retransmission to the MAC in subsequent frames due to shared request/report subslot contention.
Satellite Capacity Maps	MAC data base component providing an indication of the availability of each TDMA slot (in each channel) for assignment to a net.
Scenario	Elements of the MAC's net and TAC lists describing the current configuration of all TAC units and the manner in which the units are organized into nets.
Scenario Generator	Computer program allowing the automated creation of scenarios.
Secondary Request/Report Subslots	Additional shared request/report subslots provided by the MAC to alleviate contention in the primary (shared) request/report subslots.
Shared Subslot	One of a set of ranging or request/report subslots used in a random access mode by TAC units needing to send the MAC ranging bursts or requests, respectively.
Signalling Community	See "control community."
Signalling Data	See "system control data."
Simulated TAC	TAC unit simulated by the Call Simulator.
Slot	Basic TDMA frame sub-interval, optimized to accommodate a single 2400-bps circuit of nominal link quality.
Status Report	Data sent from a TAC unit to the MAC, in a dedicated or polled request/report subslot, specifying the TAC unit's link quality and current communications activity.
Subscriber Unit	A single TAC.
Subslot	A sub-interval of a basic (2400 bps) data slot, used for transmission of one of the kinds of system control data.
System Control Community	See "control community."
System Control Data	Refers to control burst, ranging burst, status report, and request data.
System Management Data	All data passed between major system units other than user data, i.e., control burst, ranging burst, status report, request, system status, and system operator command data.

System Operator	Principal TACS operator at the Central Control Facility monitoring/controlling system operations via the System Operator's Console.
System Operator Commands	Data sent from the System Operator's Console to the MAC, specifying commands from the system operator to reconfigure TACS.
System Operator's Console	Intecolor-8051-based console allowing the system operator to monitor/control system operations.
System Overhead Data	See "system control data."
System Status Data	Data sent from the MAC to the System Operator's Console specifying the current system status. The data are used to generate displays on the console.
System Subscriber	A single TAC unit; in some contexts a single end user.
TAC Address	Number (1-511) uniquely associated with each TAC unit.
TAC List	MAC data base component providing a detailed description of the configuration and current communications activity of each TAC unit.
TAC Unit	See "Terminal Access Controller."
Temporary Buffers	MAC buffers, one used for holding control burst messages for placement in the next control burst, and one used for holding newly arrived requests for placement in the request queue.
Terminal	A single TAC.
Terminal Access Control System	What you're reading about.
Terminal Access Controller	The communications terminal used by TACS.
Terminal Input Controller	Man/machine interfacing device employed by end users. Prompts users for request parameters, and displays status of request once it is forwarded to the MAC.
Tertiary Request/Report Subslots	Additional shared request/report subslots provided by the MAC to alleviate contention in the primary and secondary (shared) request/report subslots.
Transceiver Analysis	Process, used in the MAC assignment algorithm, of determining the slots in the frame that are currently available to a given platform for new assignments.
Two-way Circuit	See "interactive circuit."
User Data	Data generated by end users for transmission in assigned slots.
Varian	Minicomputer hosting the MAC.
Varian Call Simulator	Preliminary version of the Call Simulator, co-resident with the MAC in the Varian minicomputer.
Virtual Preemption	Function performed by the MAC assignment algorithm when trying to assign system capacity to requests: Some of the currently active circuits of precedence less than the new call precedence may be ignored, i.e., "virtually" preempted.

Voice Net

Net whose communications (user) data consist entirely of voice data.

Acronyms

AALG	See "assignment algorithm."
BIC	Buffer Interlace Controller: Varian minicomputer direct memory access controller card.
CAMS	See "Communications Area Master Station."
COMMSTA	See "Communications Area Master Station."
CPU	Computer Central Processing Unit.
CRT	Cathode Ray Tube terminal.
DAC	See "Demand Access Controller."
DEC	Digital Equipment Corporation.
DMA	Direct memory access.
FDUX	Full-duplex.
FOD	See "frame of day."
HDUX	Half-duplex.
IALG	See "input algorithm."
I/O	Input/output.
MAC	See "Multiple Access Controller."
ODT	On-line Debugging Technique: PDP-11/03 minicomputer debugging package.
PIM	Priority Interrupt Module: Varian minicomputer interrupt controller card.
RAM	Random Access Memory.
RFI	Radio-frequency interference.
RTC	Real-time clock.
S/N	Signal-to-noise (ratio).
SOF	Start of frame.
TAC	See "Terminal Access Controller."
TACS	See "Terminal Access Control System."
TDM	Time division multiplexed (multiplexing).
TDMA	Time division multiple access.
TIC	See "Terminal Input Controller."

UASC	Universal Asynchronous Serial Controller: Varian minicomputer serial (RS-232) I/O port controller card.
XSLT	Routine used to perform transceiver analysis (see "transceiver analysis").
XSUM	Routine used to perform transceiver analysis (see "transceiver analysis").

APPENDIX A

FRAME STRUCTURE DESIGN

The design of a time division multiplex frame for TACS consists of choosing two key frame parameters - specifically frame duration and the number of data slots per frame - subject to a number of constraints. Major constraints include equipment characteristics, the desire for high throughput efficiency, the need to provide adequate capacity for system control, and the requirement to support terminals equipped with half-duplex radios. From most standpoints, a longer frame duration is more desirable. The most notable exception to this rule occurs with regard to speaker-to-speaker turnaround time, which is defined to be the interval between the instant that one voice circuit user stops talking and the earliest time that he may expect to hear the response from the other party. It is of course best to minimize this time, which consists of two roundtrip earth-satellite propagation delays (total, ca. 500 ms) and approximately two frame intervals, the predominant factor. Thus the frame structure design is a tradeoff exercise, with a large number of input parameters.

A nominal user data rate of 2400 bps, corresponding to efficiently vocoded speech, is assumed throughout this design exercise. In essence, a frame structure consisting of uniform width (in time) data slots optimized for a 2400-bps user with a good link (and hence able to communicate reliably at high burst rate and with a comparatively efficient rate 3/4 code) is to be designed. Users generating data at lower baseband rates may be submultiplexed into the basic 2400-bps data slots. Subdivision of these data slots is also used to derive multiple system control circuits. On the other hand, 2400-bps users with disadvantaged links may be assigned several adjacent basic data slots, allowing them the additional time needed for communications using lower burst and/or code rates. The (strong) motivation for using a uniformly divided frame is that this structure gives a large pool of essentially equivalent data slots, which can be easily and efficiently allocated by computer-based algorithms.

Two quite different frame structures were derived for use with TACS, so that the system could work with either of two types of modems. This appendix first details the design of a frame structure appropriate for use with a fast-acquiring, 32,000-sps modem. There follows a section describing the frame structure applicable for use with the modem built into the AN/WSC-3 radio. The WSC-3 modem is limited to a burst rate of 19,200-sps and is comparatively slow to acquire since it was originally designed for continuous (i.e., non-burst-mode) operation.

Independent of which modem is available, the frame duration is restricted to a set of discrete values. For ease of implementation, it is desirable that the microprocessor performing a TAC's data buffering, a machine organized around 8-bit bytes, should be required to handle an integer number of bytes of user data during each time frame. Since 75-bps is the lowest data rate commonly

used, and since the other rates (including 2400 bps) generated by standard military equipment are all multiples of 75, it is sufficient to ensure that a 75-baud user device generates an integer number of data bytes per frame. The resultant set of possible frame durations is presented in Table A-1.

TABLE A-1
CANDIDATE FRAME DURATIONS

<u>Bytes (at 75 bps)</u>	<u>Frame Duration (s)</u>
12	1.280
13	1.387
14	1.493
15	1.600
16	1.707
17	1.813
18	1.920
19	2.027
20	2.133
21	2.240

1. Nine-slot Frame Structure (Applicable to 32-kbps Modem)

As a prelude to a step-by-step frame structure design description, some of the more important specifications affecting the design are presented. These data, included as Table A-2, reflect fundamental system parameters, equipment capabilities, and the results of a study to choose an appropriate end-to-end error correcting coding scheme.

TABLE A-2
SPECIFICATIONS INFLUENCING FRAME FORMAT FOR USE WITH 32 KSPS MODEM

	<u>Symbol</u>	<u>User Data Burst</u>	<u>Control or Request/Report Burst</u>	<u>Ranging Burst</u>
Channel symbol rate	C	32 kbps	9.6 kbps	9.6 kbps
User data rate (nominal)	U	2.4 Kbps	N/A	N/A
Code rate	R	3/4	1/2	1/2
Code constraint length (flush bits)	L	9 bits	7 bits	7 bits
Interleaver block length	I	256 bits	N/A	N/A
PN synch word length	P	64 symbols	64 symbols	64 symbols
Guard, modem acquisition, and R/T settling time	A	0.015 s	0.015 s	0.054 s

The defined symbols correspond to the specification for a user data burst, and are used in the formulas which follow. The system control (i.e., control, request/report, and ranging burst) data are transmitted at low code and burst rates, in order to ensure robustness. Slots designed to accommodate ranging bursts must have large timing guard bands in order to handle the ± 20 -msec

initial range ambiguity. Interleaving is not required on system control data, since the expected RFI pulse duration is comparable to the duration of a single channel symbol. The advantage of not interleaving system control data is that it allows transmission of system control data bursts of length other than a multiple of 256 symbols - the interleaver block size. The result is a large savings in overhead, especially for request/report and ranging bursts, which can be as short as 110 and 38 symbols, respectively.

The first task in frame structure design is to determine the number of data slots (N) to be derived from each frame. A simple upper bound on this buffering compression factor is:

$$N \triangleq \frac{\text{channel burst rate}}{\text{user data rate}} = \frac{C \times R}{U} = \frac{32000 (3/4)}{2400} = 10.0 \quad (\text{A-1})$$

Clearly, when the necessary slot overhead (i.e., for guard bands) is factored in, only nine slots can be offered. Although a frame of fewer than nine slots would give shorter frame duration and (hence) speaker turnaround time, the nine-slot frame is chosen for its maximization of throughput efficiency.

Given the number of data slots per time frame and the per-slot guard band, preamble, and coding requirements, a minimum frame duration (T) may be calculated. Inequality (A-2) quantifies the real constraint that a full frame of user data plus all associated overhead must fit in a single time slot.

$$\left\lceil \frac{(U \times T + L)/(R \times I)}{C} \right\rceil \times I + A + \frac{P}{C} \leq \frac{T}{N} \quad (\text{A-2})$$

The notation $\lceil \cdot \rceil$ signifies the smallest integer greater than the value enclosed and reflects the need to handle user data in full interleaver blocks. Substituting from Table A-2 and Eq. (A-1) gives:

$$\frac{\lceil (2400T + 9)(4/3)/256 \rceil 256}{32000} + 0.015 + \frac{64}{32000} \leq \frac{T}{9} \quad (\text{A-3})$$

The minimum value for T from Table A-1 which satisfies the inequality is 1.813 seconds.


At this point, it is necessary to check how effectively the basic data slots may be subdivided into system control circuits, and to determine the amount of capability the ensuing frame structure provides for a half-duplex radio equipped terminal. Specific issues include:

- a. the number of request/report subslots (at 48 bits, uncoded, per request/report) which can be derived from one basic data slot, and efficiency with which the subslots fill that data slot.
- b. the total amount of control burst capacity provided (minimum required: 300 bits).

- c. the requirement that a half-duplex terminal should be capable of continually monitoring the control subslot, accessing any one of the report/request subslots, and working two two-way data circuits, all on a noninterfering basis.

These considerations force frame duration upward two steps, to a value of 2.027 s. The chosen nine-slot frame structure is depicted in Fig. A-1. In addition, Table A-3 indicates the total time duration for each type of burst, and the efficiency with which the bursts pack into the 225-ms basic data slots.

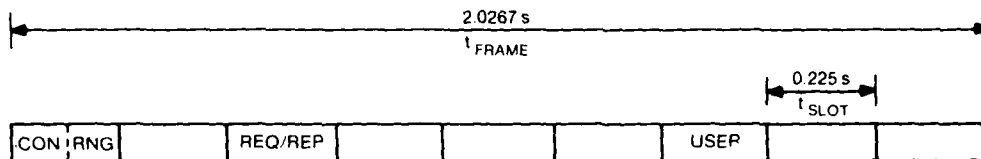
TABLE A-3
DATA AND SYSTEM CONTROL BURST DURATIONS AND PACKING EFFICIENCIES
(Nine-Slot Frame)

	<u>Control</u>	<u>Ranging</u>	<u>Request/Report</u>	<u>User Data</u>
Guard and modem acquisition:	0.015 s	0.054	0.015	0.015
Burst synch:	0.007	0.007	0.007	0.002
Data:	+ 0.072 (336 bits)	0.004 (12 bits)	0.012 (48 bits)	0.208
Total per burst:	0.094	0.065	0.034	0.225
Bursts per basic data slot:	<u>x 1</u>	<u>x 2</u>	<u>x 6</u>	<u>x 1</u>
	0.094	0.130	0.204 s	0.225 s
<div style="text-align: center;">  0.224 s </div>				

This frame structure provides adequate system control circuit capacity, and allows a half-duplex radio equipped terminal to support two simultaneous, independent, two-way data circuits in a quite flexible manner. Under comparatively good ("nominal") link conditions when a single basic data slot per circuit is required, the half-duplex terminal can use any of six pairs of slots to provide its two independent circuits. The slots must be separated by at least 280 ms, or effectively, by two basic data slots. Even in the case of slightly degraded link conditions, when a lowered burst or code rate necessitates two basic data slots per circuit, the half-duplex terminal can maintain two independent circuits, using slot pairs 4-5 and 8-9. A full-duplex radio equipped terminal is, of course, not required to maintain 280 ms between accesses in the frame, so such a terminal has fully independent use of all basic data slots. Indeed, even when link conditions are degraded to the point where three basic data slots are required per circuit, the full-duplex terminal can support a pair of independent, simultaneous circuits. Under good link conditions, the full-duplex terminal can simultaneously work four data circuits - equal to its complement of I/O ports.

2. Five-slot Frame Structure (Applicable to 19.2 ksps WSC-3 Modem)

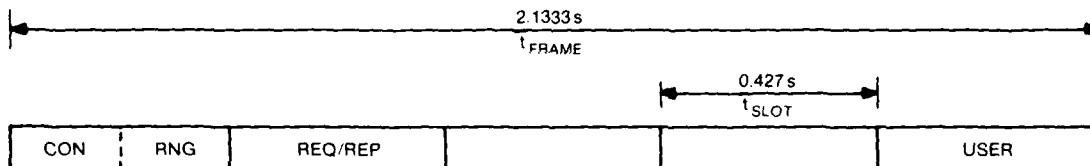
Specifications influencing this frame structure design are presented in Table A-4.



106775-N

CON: CONTROL SUBSLOT
 RNG: RANGING SUBSLOTS (2)
 REQ/REP: REQUEST/REPORT SUBSLOTS (6)
 USER: 2400 bps USER DATA SLOT

Fig. A-1. Nine-slot frame structure.



105463-N

CON: CONTROL SUBSLOT
 RNG: RANGING SUBSLOTS (3)
 REQ/REP: REQUEST/REPORT SUBSLOTS (7)
 USER: 2400 bps USER DATA SLOT

Fig. A-2. Five-slot frame structure.

TABLE A-4
SPECIFICATIONS INFLUENCING FRAME FORMAT FOR USE WITH 19.2 KSPS MODEM

	<u>User Data Burst</u>	<u>Control or Request/Report Burst</u>	<u>Ranging Burst</u>
Channel symbol rate	19.2 kbps	9.6 kbps	9.6 kbps
User data rate (nominal)	2.4 kbps	N/A	N/A
Code rate	3/4	1/2	1/2
Code constraint length (flush bits)	9 bits	7 bits	7 bits
Interleaver block length	256 bits	N/A	N/A
PN synch word length	64 symbols	64 symbols	64 symbols
Guard, modem acquisition and R/T settling time	0.040 s	0.040 s	0.079 s

Equation (A-1) gives an upper bound data compression factor of 6.0, so with allowance for overhead a frame format of five uniform data slots is chosen. The minimum frame duration from Table A-1 which satisfies inequality (A-2) is 1.493 s. The six-step higher frame duration of 2.133 s is chosen, primarily to allow multiple simultaneous two-way circuits for half-duplex terminals.

The resultant five-slot frame structure is depicted in Fig. A-2. Table A-5 shows total durations for the various classes of bursts, and also indicates the way in which system control data bursts submultiplex into the 427-ms basic data slots.

TABLE A-5
DATA AND SYSTEM CONTROL BURST DURATIONS AND PACKING EFFICIENCIES
(Five-slot Frame)

	<u>Control</u>	<u>Ranging</u>	<u>Request/Report</u>	<u>User Data</u>
Guard and modem acquisition:	0.040 s	0.079	0.040	0.040
Burst synch:	0.007	0.007	0.007	0.004
Data:	+ 0.095 (448 bits)	0.004 (12 bits)	0.012 (48 bits)	0.360
Total per burst:	0.142	0.090	0.059	0.404
Bursts per basic data slot:	<u>x 1</u>	<u>x 3</u>	<u>x 7</u>	<u>x 1</u>
	0.142	0.270	0.413 s	0.404 s
<hr/>				
	0.412 s			

This frame structure offers somewhat more system control capacity than the nine-slot frame, but due to its lower burst rate capability provides terminals with a bit less data circuit flexibility. Full-duplex radio equipped terminals may work three simultaneous, independent circuits under nominal link conditions, but must reduce to a single circuit if links degrade. Similarly, terminals with half-duplex radios may support two simultaneous circuits (slots 3 and 5) when links are good and otherwise may support a single access.

APPENDIX B
(EXAMPLE OF) CIRCUIT ESTABLISHMENT/TERMINATION TIMING

This appendix has been included in order to give the reader a system-level view of the process of establishing and terminating circuits. The process is described via the example diagrammed in Fig. B-1. The figure shows the status of, or actions taken by, various TACS units during the servicing of a request for an assignment of three frames duration. Time is displayed from the call initiator's final TIC keypad entry (for entering request parameters) through the deletion of the resulting assignment from all parts of the system. The example assumes request origin at a TIC connected to the remote TAC. (The timing relationships for requests from the Master TAC and the Call Simulator are slightly different.) No forms of request delay or blockage (request/report subslot contention, insufficient satellite or transceiver capacity for assignment, delays in the request queue, etc.) are included in the example. Termination of the circuit is shown via expiration of the request's (three frame) duration estimate rather than by preemption or user hangup.

The example of Fig. B-1 spans eight TDMA frames, during which the status of the user's TIC and I/O port, and of the System Operator's Console is shown. Also shown are all satellite, Master TAC, and MAC activities relating to the request and the subsequent assignment. A brief description of the example follows.

When the remote TAC unit has received all of the parameters needed to characterize a request from the end user's TIC, the request is formatted and prepared for transmission in one of the request/report subslots in the next TDMA frame (frame 2 in this example). The Master TAC receives the request data and performs a parity check. If no parity errors are detected, the request's format is modified and it is passed to the MAC (along with any local Master TAC requests) at the beginning of TDMA frame 3. The MAC then processes the request, reflecting the resulting circuit assignment in (1) its data base, (2) the next control burst, and (3) the next system status data stream to be sent to the System Operator's Console. The control burst data are forwarded to the Master TAC at the end of frame 3. The Master TAC inserts parity check bits in the control burst data and prepares the data for transmission in the control subslot of frame 4. When the remote TAC receives the control burst it performs a parity check on the data and, if no errors are found, processes the control burst contents. Recognition of the circuit assignment message in the control burst generates a "circuit assigned" display at the call initiator's TIC, followed by a "(net name) active" display in succeeding frames. The user's I/O port is activated so that the data buffering and bursting functions may begin. The data are burst according to the parameters specified by the MAC in the assignment message (channel, slot, burst rate, code rate, etc.), the first occurring in a data slot in frame 5. At roughly the same time as the assignment is recognized and implemented by the remote TAC (frame 4), it appears in the System Operator's Console displays.

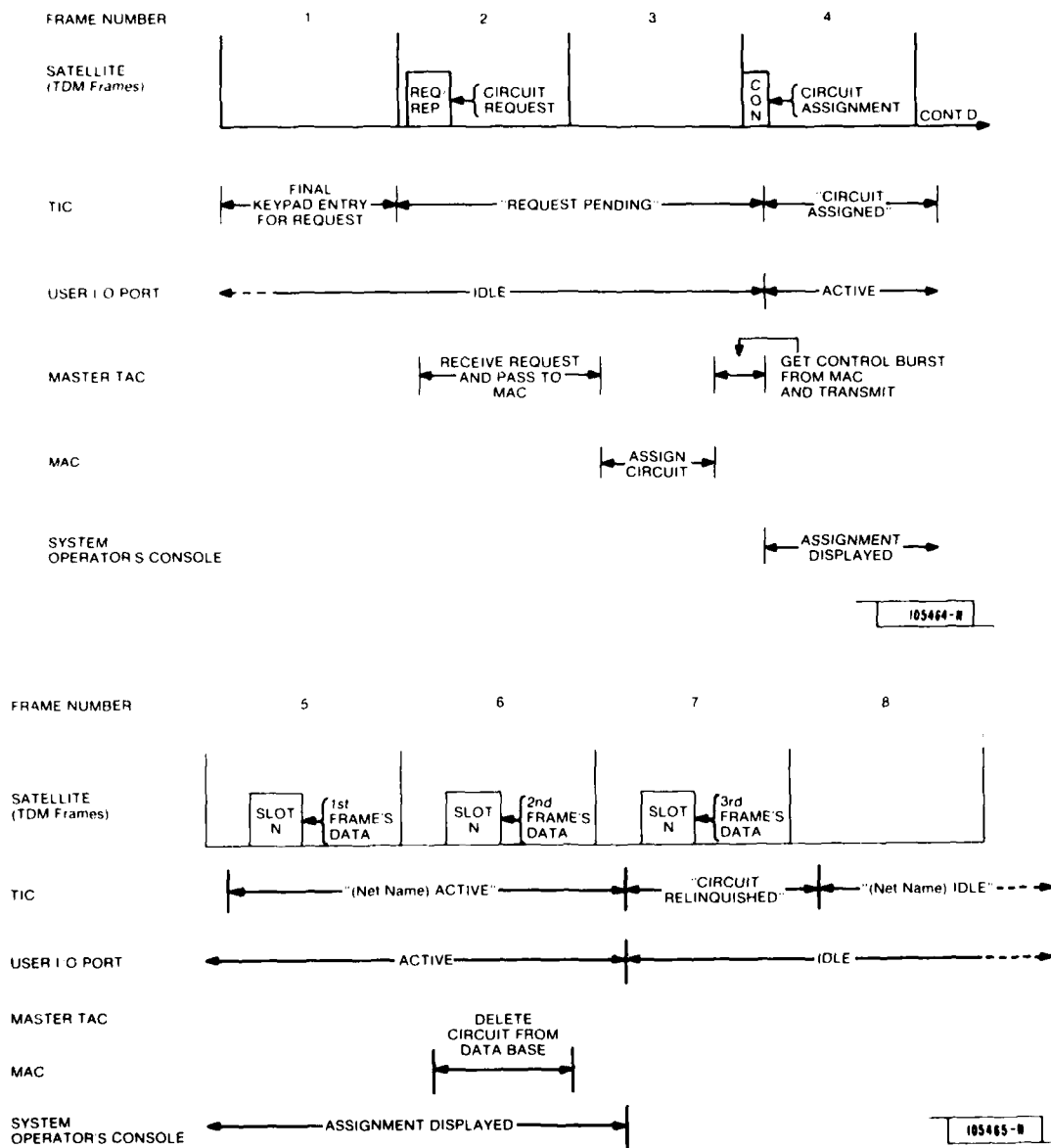


Fig. B-1. Timing of the establishment and termination of a three frame assignment.

Circuit termination (for this example) is a simpler process. If the call initiator had specified a call duration of three frames, the MAC deletes the circuit assignment from all parts of its data base during its processing activity for frame 6. (It had originally made the assignment during frame 3 processing.) As this deletion is reflected in the MAC's output to the System Operator's Console, the assignment no longer appears on the console's displays generated during frames 7, 8,.... The remote TAC independently terminates the circuit at the beginning of frame 7 (using its knowledge of the initial frame of the assignment and its specified duration). "Circuit relinquished" is displayed on the call initiator's TIC, followed by a "(net name) idle" display in succeeding frames. The third, and final, data burst occurs in frame 7, and the end user's I/O port is disabled. At this point the end user can generate a new request.

APPENDIX C
SYSTEM MANAGEMENT DATA FORMATS

System management data refers to all of the data passed between the major system units (MAC, Call Simulator, System Operator's Console, Master TAC, and remote TAC) other than the end user data transmitted via MAC-assigned satellite circuits. All system management data are ultimately processed or generated by the MAC. (The MAC has no interface with the end user data.) The data are passed between units via satellite (in specially designated TDMA subslots) and/or hard-wired links. The data include: (1) the control burst data sent from the MAC to the Call Simulator and to the real TAC units; (2) status report, request, and ranging burst data sent from the real and simulated TAC units to the MAC; (3) system status data sent from the MAC to the System Operator's Console; and (4) system operator commands sent from the System Operator's Console to the MAC.

The formats of the system management data streams are described in the following sections. As the TACS Central Control Facility units communicate via serial interfaces which are (eight-bit) byte oriented, all formats are displayed byte-wise. As with the MAC data base (see Appendix E), most of the data entries are explicit, with no clever coding used to "save bits." Also of note is the overhead provided in the system management data streams, both as unassigned "spare" bits, and as bits assigned to features that have not been fully developed in the current TACS implementation. These features, which would be useful in a full-scale system, will be pointed out in the data stream descriptions.

1. Control Burst Data

Control burst data are assembled by the MAC and are sent directly to the Call Simulator and Master TAC. The Master TAC transmits the control burst to the remote TAC in the control subslot of the TDMA frame. A byte-wise parity-check scheme has been implemented to provide control burst data error protection beyond that already provided by the rate 1/2 convolutional code. (Such a scheme is easy to implement on the byte-oriented Intel 8080 part of the TAC units.) As shown in Fig. C-1, the 42 bytes of the control burst are divided into six seven-byte parity-check blocks, each block containing six bytes of data and one parity byte. The MAC assembles all of the control burst other than the parity bytes. The Master TAC unit inserts the parity bytes (using "column" parity for each block) before encoding, interleaving, and transmitting the control burst. The remote TAC uses these bytes for a parity check after the control burst has been acquired, de-interleaved, and decoded. Any block failing the parity test is discarded.

Figure C-1 shows the control burst to be divided into an "inelastic" and an "elastic" area. The format of the data in the inelastic area does not change from frame to frame. The 27 data bytes of the elastic area, however, are used

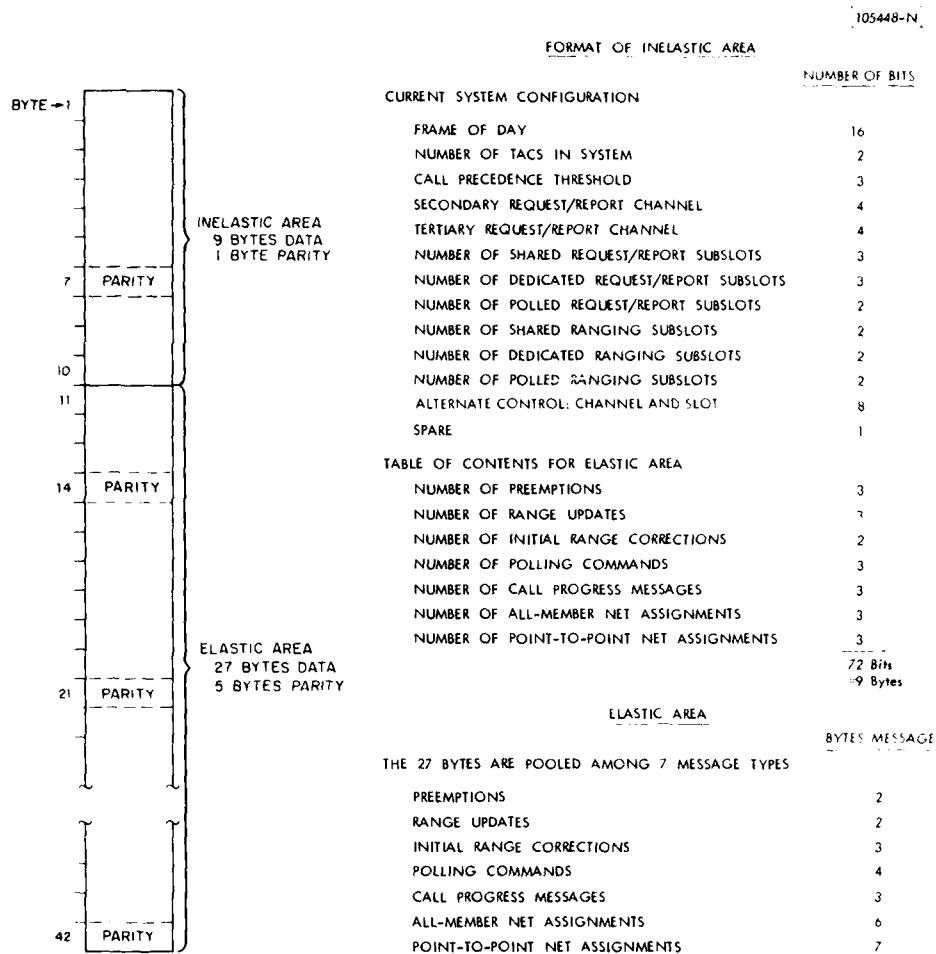


Fig. C-1. General control burst data format.

in a pooled fashion for seven types of control burst messages. The number of messages of each type varies from frame to frame depending on the results of MAC processing. This feature allows for efficient use of control burst capacity.

The nine data bytes of the inelastic area are divided between fields used to inform TAC units of the current system configuration, and a "table of contents" defining the number of messages of each type in the elastic area. The system configuration fields contain both static and dynamically changing data. The "frame of day" and "number of TACS in system" codes are used for system synchronization functions such as the automatic termination of circuits whose time estimates have expired, and the cycling of dedicated request/report subslots through the user community. The call precedence threshold (not currently implemented) informs the TAC units of the minimum precedence level required for new requests. High thresholds would be invoked by the MAC and/or the system operator in cases of heavy system loading. The secondary and tertiary request/report channel fields inform TAC units of the availability of request/report subslots in channels other than the primary (control burst) channel. Although the request/report subslot configuration currently remains static (determined at system initialization), it could be dynamically managed by the MAC and/or the system operator. The next three fields define how the available request/report subslots are divided among dedicated subslots used for status reports and requests, shared (random access) subslots used for requests only, and polled subslots used for status reports and requests. These fields are followed by three fields similarly indicating the use of the ranging subslots. (Note that the MAC does not currently send out polling commands in the elastic area of the control burst, hence the polled request/report and ranging subslot features are not implemented. These subslots would be assigned for one frame only to the TAC units receiving the polling commands. Also unimplemented are dedicated ranging subslots. These would regularly cycle through the subscriber unit community in a fashion similar to the currently implemented dedicated request/report subslots.) The final field of the system configuration data specifies the location (channel and slot) of an alternate control subslot being used by TAC units belonging to an alternate control community ("B" community). Such a two community system can make more efficient use of satellite resources than a single control community. Although the MAC's real-time and processing structure does not presently generate such an alternate control burst (or handle alternate request, status report, or ranging burst input), MAC software is capable of dynamically accounting for the satellite and transceiver capacity required for a (simulated) alternate community.

The seven types of control burst messages that may be found in the elastic area of the control burst are: preemptions, range updates, initial range corrections, polling commands (not implemented), call progress messages, all-member net circuit assignments, and point-to-point net circuit assignments. The format of each of these messages is given in Fig. C-2.

105449-N

	NUMBER OF BITS		NUMBER OF BITS
PREEMPTION		CALL PROGRESS MESSAGE	
CHANNEL	4	CALLING TAC ADDRESS	9
SLOT	4	NET ADDRESS	7
SUBSLOT	3	MESSAGE CODE	4
DELAY	3	SPARES	4 (3 bytes)
SPARES	2 (2 bytes)		
RANGE UPDATE		ALL-MEMBER NET ASSIGNMENT	
TAC ADDRESS	9	CALLING TAC ADDRESS	9
CORRECTION	5	NET ADDRESS	7
SPARES	2 (2 bytes)	CHANNEL	4
INITIAL RANGE CORRECTION		SLOT	4
TAC ADDRESS	9	SUBSLOT	3
CORRECTION	10	NUMBER OF SLOTS	3
SPARES	5 (3 bytes)	CODE RATE	2
POLLING COMMAND		BURST RATE	2
TAC ADDRESS	9	CALL PRECEDENCE	3
COMMAND CODE	3	DELAY	3
CHANNEL	4	CALL DURATION	5
SLOT	4	BROADCAST/TWO-WAY CIRCUIT	1
SUBSLOT	3	SPARES	2 (6 bytes)
DELAY	3		
SPARES	6 (4 bytes)	POINT-TO-POINT NET ASSIGNMENT	
		ALL-MEMBER NET PARAMETERS	46 (As Above)
		SPARE	1
		CALLED TAC ADDRESS	9 (7 bytes)

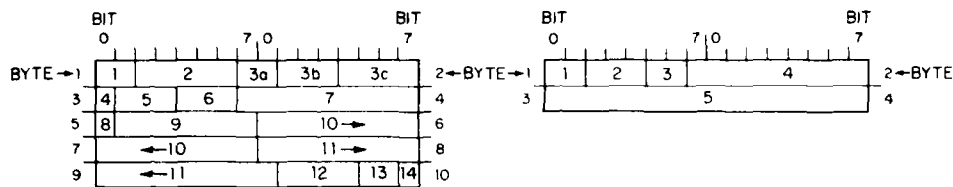
Fig. C-2. Control burst message format.

Preemption messages instruct all TAC units to immediately terminate any communications actively using the channel and beginning slot specified. Preemption messages are always processed by TAC units before assignment messages. Subslot level preemptions and delayed preemptions are not implemented. Range update messages send signed satellite range (timing) corrections to subscriber units that have just sent status reports to the MAC. The range corrections reduce the addressed TAC's transmit timing uncertainty from the duration of a TDM data slot guard interval (guard band) to the quantization level of the correction. (The quantization is on the order of the time required to burst a single channel symbol.) Similarly, initial range correction messages are issued to TACs whose ranging burst data have just been received at the MAC, reducing transmit timing uncertainty from ± 20 msec to the quantization level of the correction. For both of these messages, the range corrections are derived from the arrival times of (ranging or status report) bursts at the Master TAC.

All parts of the control burst data (both inelastic and elastic) described up to this point can be classified as system control data. The last three control burst message types provide call (request) response information to TAC units. Call progress messages are similar to commercial busy signals. They are addressed (via TAC and net addresses) to the particular end user that has made an "unsuccessful" request, and contain a code that is used to generate a display at the user's Terminal Input Controller (TIC). This display gives the end user an indication of why the MAC could not honor the request with a circuit assignment. At this point, the user may attempt a new call. Figure 2-7 lists the call progress messages that have been implemented. Circuit assignment messages are the result of "successful" requests. Two kinds of messages are provided for the two kinds of TACS communications nets. All-member net assignments are implicitly addressed to every TAC with an I/O port belonging to the all-member net from which the request originated. Point-to-point net assignments are explicitly addressed to specific called and calling parties within the net (and implicitly to the net controller, if any). In either case, circuit assignment messages contain all of the information required to set up the new circuit: the channel, first slot, and number of slots to be occupied by the circuit; the burst and code rates needed to provide adequate error protection; and the precedence, duration, and direction (two-way or broadcast) of the assignment. (The last three parameters are supplied by the end users when generating requests.) Subslot level assignments and delayed assignments are not implemented.

2. Status Report, Request, and Ranging Burst Data

Figure C-3 shows the formats for blocks of status report, request, and ranging burst data as they are sent to the MAC by the Master TAC and Call Simulator. The MAC is capable of inputting and processing data streams consisting of many of these blocks ordered in any fashion. (The limits currently imposed are 80 bytes/frame for data from the Master TAC, and 200 bytes/frame



STATUS REPORT/REQUEST BLOCK

KEY TO FIELDS

- (1) BLOCK TYPE
- (2) CALL DURATION
- (3) (a) PERCENT RFI
- (b) DOWNLINK S/N
- (c) V/O PORT ACTIVITY
- OR
- (3a)-(c) CALLING PARTY'S TAC ADDRESS
- (4) BROADCAST/TWO-WAY CIRCUIT
- (5) CALL PRECEDENCE
- (6) NUMBER OF PREVIOUS REQUEST ATTEMPTS
- (7) CALLED PARTY'S TAC ADDRESS
- (8) FLAG FOR: LATE ENTRY TO NET
- OR CONFERENCE CALL
- (9) NET ADDRESS
- (10) TIME OF ARRIVAL IN FRAME
- (11) UPLINK QUALITY DATA
- (12) REPORT/REQUEST ORIGIN
- (13) REPORT/REQUEST CHANNEL
- (14) SPARE

RANGING BURST BLOCK

KEY TO FIELDS

- (1) BLOCK TYPE
- (2) NUMBER OF PREVIOUS RANGING ATTEMPTS
- (3) SPARES
- (4) RANGING TAC'S ADDRESS
- (5) TIME OF ARRIVAL IN FRAME

Fig. C-3. Status report, request, and ranging burst formats.

for data from the Call Simulator.) Requests and status reports generated at the remote TAC are transmitted to the Master TAC via satellite in the request/report subslots. The data transmitted consist of bytes 1 through 5 of the status-report/request format in Fig. C-3, plus one byte of "column" parity. The Master TAC uses the parity byte to perform a parity check on any data received in the request/report subslots. This parity-check scheme, similar to the control burst parity-check scheme, provides request/report subslot error protection beyond that already provided by the rate 1/2 convolutional code. Any status report or request showing a parity error is discarded by the Master TAC. Otherwise, the block of data is forwarded to the MAC with the parity byte removed and the information in bytes 6 through 10 appended. Requests and status reports generated locally at the Master TAC and by the Call Simulator are sent directly to the MAC in the ten-byte format. Ranging bursts are only generated by the remote TAC (at remote TAC startup time). As with status reports, requests, and control bursts, ranging bursts are transmitted (in the ranging subslots) with parity appended for added error protection. Figure C-3 shows the format of ranging burst data as received by the MAC, i.e., after it has been acquired, parity-checked, and reformatted by the Master TAC.

The next few paragraphs' discussion of the status report and request data formats assumes the case of data originating at the remote TAC unit. Since, in a full-scale TACS, most TAC units would be "remote," i.e., linked to a Central Control Facility via satellite only, the path taken by requests and status reports generated at the remote TAC is most representative of system operation. Although the requests and reports originating at the Master TAC, and those generated by the Call Simulator, are not transmitted over the satellite, the data forwarded to the MAC by these units are formatted in exactly the same fashion as the remote TAC's data. The great majority of the MAC's functions make no distinction between simulated and real subscriber units, making the MAC's processing burden and the overall system performance virtually identical to that which would be obtained in a full-scale system.

The data sent in the request/report subslots, and hence the request/report blocks sent to the MAC, have been formatted such that a single block can represent a status report, a request, or both. Field (1) of the request/report format in Fig. C-3 contains a code defining the kind of data to be found in the block. Fields (2) and (4) - (9) are only used when the block contains a request. The call duration (time requirement for a circuit specified in frames for record nets and minutes for voice nets), broadcast/two-way circuit flag, call precedence, called TAC address (for point-to-point nets only), and late entry to net (all-member nets) or conference call (point-to-point nets) flag are generated by the TACS end user making the call. The number of previous request attempts in field (6) is automatically incremented by the end user's TAC unit every time it must retransmit a request due to contention in a shared subslot. The net address in field (9) is also automatically inserted by the TAC unit. Blocks that are indicated by the code in field (1) as containing only a request

are transmitted in the shared request/report subslots. In this case fields (3a)-(c) contain the address of the calling party's TAC. This is required in order for the MAC to know which TAC has made use of the shared subslot. Subslots that are dedicated each frame to specific TAC units (these regularly cycle through the entire TAC community) are used for status reports, which may or may not be combined with requests. In this case, the reporting/calling TAC's address is implicit in the subslot used for data transmission. Hence fields (3a)-(c) are used for status report data: (3a) contains a code representing the percentage of the time RF1 has been present at the TAC unit's receiver, (3b) codes the unit's estimate of its downlink signal-to-noise ratio, and (3c) contains a one-bit flag for each of the TAC's four I/O ports, indicating whether or not the port is currently engaged in communications.

Fields (10)-(13) contain data appended to both status reports and requests by the Master TAC. The "time of arrival in frame" (acquisition time of a request or report at the Master TAC) and the "report/request channel" fields are used by the MAC to determine the subslot in which a status report or request was broadcast. (The report/request channel field differentiates the data received in the primary, secondary, and tertiary report/request channels.) The MAC requires knowledge of the subslots used for two reasons: (1) The identification of a status report with a given TAC is determined implicitly from the (dedicated) subslot used for the report, and (2) the MAC must, in the current system, simulate contention for the shared request/report subslots, ignoring the data associated with any subslot used by multiple real and/or simulated TACs. The time of arrival in the frame is also used by the MAC to send satellite range updates to the TAC units that have just issued status reports. Field (11) is reserved for an estimate of the uplink quality of the subscriber unit making the request or report. This feature is not implemented in the present TACS. Uplink quality estimates would be derived from the demodulation or decoding of the first six bytes of request/report data blocks at the Central Control Facility. The "report/request origin" field is used by the MAC software in the few places where it must distinguish between data arriving from the remote TAC, the Master TAC, and the simulated TACs. (This field is also used by the MAC to make a further distinction between simulated TACs that are "remote" and a simulated CAMS with nine TAC units.) This field would not be required in a full-scale system.

The ranging burst data fields in Fig. C-3 are similar to the report/request data fields. The code in field (1) identifies the block to the MAC as four bytes of ranging burst data. Field (2) indicates the number of times the ranging burst had to be retransmitted due to contention in the shared ranging subslots. Field (4) specifies the address of the ranging TAC. Field (5), as with request/report field (10), is used to determine the ranging subslot used for transmission of the block of data, and to issue initial satellite range corrections to TACs that have just sent the MAC ranging bursts.

3. System Status Data

A stream of system status data is sent from the MAC to the System Operator's Console during every frame. The data are used at the console to generate displays of system configuration and activity. The format and quantity of the data sent from the MAC are invariant to which display has been selected, or any commands that the system operator has forwarded to the MAC. As different displays are selected, different parts of the system status data stream are accessed by the Operator's Console.

The system status data stream, diagrammed in Fig. C-4, is 758 bytes long. Its major sections describe the current circuit assignments (648 bytes), contents of the MAC's request queue (81 bytes), and activity in the ranging and request/report subslots (15 bytes). Also sent are the frame of day, header bytes used for I/O synchronization between the MAC and the Operator's Console, and spares.

The format of the system status data describing circuit assignments is nearly identical to the MAC's "assignment list." The assignment list gives detailed information on the usage of every TDM slot in every satellite channel. Indication is given as to whether the slot is being used for system overhead (e.g., for the control subslot), is presently assigned to a communications net, is awaiting such assignment, or is not in the demand assignment pool. Appendix E contains a detailed description of the assignment list. The only difference between the format of the MAC's assignment list and the format of the circuit assignment data sent to the System Operator's Console is that, in the latter case, a special code is placed in the data block describing any TDMA slot in which a preemption has just occurred. This allows the generation of preemption displays at the console.

Part of the system status data stream indicates the status of the MAC's request queue. A four-byte block of data is placed in the stream for each request that was in queue at the beginning of the current frame. (The system status data stream has enough capacity to hold the first 20 requests found in queue each frame.) Blocks are assembled both for requests that arrived at the MAC during the current frame, and for requests that had been held over in queue for one or more frames. The blocks are ordered in the same manner as they were serviced by the assignment algorithm (primarily by precedence, and secondarily by frame of arrival). The blocks, formatted as shown in Fig. C-5, contain the following entries, taken directly from the corresponding request queue entries (see Appendix E for a description of the MAC's request queue): the calling party's TAC address, broadcast/two-way circuit flag, call precedence, number of previous request attempts, late entry to net or conference call flag, and net address. There are two additional entries, one indicating the number of frames the request has been held over in queue awaiting service, and the other indicating the action, if any, taken on the request in the current frame. For requests that have just received final action, the second of these entries

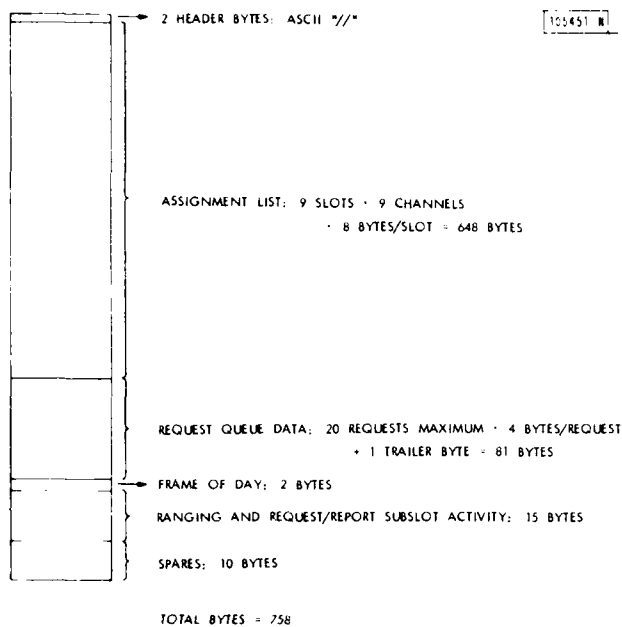
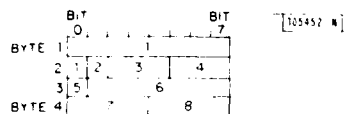


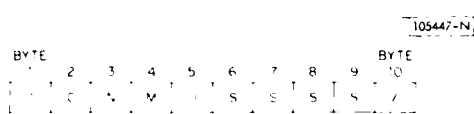
Fig. C-4. System status data stream.



KEY TO FIELDS

- (1) CALLING PARTY'S TAC ADDRESS
- (2) BROADCAST/TWO-WAY CIRCUIT
- (3) CALL PRECEDENCE
- (4) NUMBER OF PREVIOUS REQUEST ATTEMPTS
- (5) FLAG FOR: LATE ENTRY TO NET
OR CONFERENCE CALL
- (6) NET ADDRESS
- (7) NUMBER OF FRAMES IN QUEUE
- (8) ACTION TAKEN ON REQUEST

Fig. C-5. System status data stream request queue block format.



KEY

- / - HEADER AND TRAILER BYTES (ASCII "//")
- C - COMMAND CODE
- (1) DELETE SPECIFIED CHANNEL FROM DEMAND ASSIGNMENT POOL
 - (2) ADD SPECIFIED CHANNEL TO DEMAND ASSIGNMENT POOL
 - (3) PREEMPT ASSIGNMENT TO SPECIFIED NET IN SPECIFIED CHANNEL AND SLOT
- N = CHANNEL NUMBER
- M = SLOT NUMBER (For Command Code 3 Only)
- O = NET ADDRESS (For Command Code 3 Only)
- S = SPARES

Fig. C-6. System operator command format.

contains either the call progress message code (see description of control burst data in Appendix C.1) sent to the call initiator, or a code indicating a circuit assignment. These requests will be removed from the queue. The "action taken" entry also indicates all requests that could not yet be serviced as "left in queue." These requests will appear again in the next frame's system status data stream.

The last major part of the system status data stream can be used to inform the system operator of the usage of the current frame's ranging and request/report subslots. It includes two types of information: (1) the current subslot configuration and (2) the activity in each subslot. The current subslot configuration specifies the number of request/report subslots available to TAC units, i.e., whether or not there is a secondary and/or tertiary request/report channel. It also specifies which of the ranging and request/report subslots are being used in a shared, dedicated, or polled mode. Although the current TACS implementation does not dynamically change the subslot configuration, this feature would be useful in a fully operational system. "Activity in each subslot" refers to information on the number of TAC units, real and/or simulated, that have just transmitted in the subslot. Only shared subslots should show contention, i.e., attempted usage of the subslot by multiple TAC units. Knowledge of such multiple attempts in a subslot is a byproduct of the MAC's subslot contention simulation. Included with the subslot usage data is a count of the number of requests and status reports that were received from TAC units at the central control site in the current frame. The full-scale central control site exists in simulation only (in the Call Simulator), and contains nine TAC units whose requests and status reports are not involved in the MAC's subslot contention simulation. (The TACs at a real central control site would communicate with the MAC via a hard-wired link.)

4. System Operator Commands

Commands generated at the System Operator's Console keyboard are checked for validity, coded, and sent to the MAC within one frame. A minimal number of bytes are necessary to specify the command to the MAC. (Ten bytes are currently sent to the MAC, including header, trailer, and spares.) Each command consists of an identifying code plus any necessary arguments (such as channel, slot, and net). The command format is given in Fig. C-6. Although commands from the system operator do not alter the format of system status data sent to the Operator's Console, they do reconfigure TACS, this fact being reflected in the content of the system status data.

APPENDIX D
FIVE- AND NINE-SLOT SYSTEM IMPLEMENTATION DIFFERENCES

As mentioned in the introduction to this report, TACS has been designed to operate with either of two different TDMA frame structures: the "nine-slot" and "five-slot" frames. The nine-slot frame structure presupposes TAC units that are equipped with modems capable of operating at 32-ksps; 19.2-ksps operation is required by the five-slot structure.

Appendix A details the design of the two frame structures. The major difference between the structures is, of course, the number of "basic data slots" per frame: The nine-slot structure, by virtue of the higher burst rate on which it is based, can provide (for satellite links of nominal quality) much greater throughput than the five-slot case. The five-slot frame, however, provides a greater amount of system control data capacity: seven (vs six) request/report subslots per basic data slot, three (vs two) ranging subslots per basic data slot, and 448 (vs 336) bits of control burst data per frame. In addition, the two TDMA frames are of slightly different durations: 2.027 s for the nine-slot case and 2.133 s for the five-slot case.

The desire to operate TACS using either of two significantly different frame structures has impacted the design of the software for each of the Central Control Facility units described in this report. Each unit requires either two versions of its program (one for each frame structure) or a single program that performs the appropriate tests and branches based on the frame structure currently in use. All other major sections of, and appendices to, this report present implementation details for the case of nine-slot system operations (except where explicitly noted to the contrary). The following subsections (D.1 through D.4) of this appendix present, for the MAC, the Call Simulator, the System Operator's Console, and the Scenario Generator, respectively, the major differences between five- and nine-slot frame software.

Before detailing the unit by unit design differences, a few system-level differences between the five- and nine-slot implementations will be noted. First, although MAC software is fully capable of servicing real, as well as simulated, TAC units during nine-slot operations, the real TAC units are only used during five-slot operations. This is because the real subscriber units can currently handle a maximum burst rate of 19.2-ksps. During either five- or nine-slot operations that involve only simulated TAC units, a Syntronics frequency synthesizer (model SI-70) and a signal-conditioning box are used to provide the start of frame interrupt pulse that would otherwise be provided by the real Master TAC unit.

Other system-level differences are in the amount of control burst data passed from the MAC to the Call Simulator and Master TAC, and the amount of system status data passed from the MAC to the System Operator's Console. As the five-slot structure provides 112 bits (14 bytes) more control burst

capacity per TDMA frame than does the nine-slot structure, this extra capacity is added to the control burst elastic area (see Fig. C-1) in the form of two more parity-check blocks. The system status data stream sent to the System Operator's Console (see Fig. C-4) is 288 bytes shorter in the five-slot system than in the nine-slot system. This results from a shorter MAC assignment list; only 45 TDMA slots need be described by assignment list blocks (vs 81 slots for the nine-slot frame).

A final system-level difference between the five- and nine-slot frame structures is in the kind of link adaption provided. Although the mechanisms of link adaption are identical, the burst and code rate combinations (and the resulting number of contiguous TDMA slots) that are used for circuits with nominal or degraded links are different in the five- and nine-slot systems. These are tabulated in Table D-1.

TABLE D-1
BURST/CODE RATE COMBINATIONS, AND RESULTING NUMBER OF CONTIGUOUS
SLOTS FOR ASSIGNMENTS IN FIVE- AND NINE-SLOT SYSTEMS

<u>Five-Slot Frame</u>			<u>Nine-Slot Frame</u>		
Burst Rate (kbps)	Code Rate	Number of Contiguous Slots	Burst Rate (kbps)	Code Rate	Number of Contiguous Slots
* 19.2	3/4	1	* 32.0	3/4	1
19.2	1/2	2	19.2	3/4	2
9.6	1/2	3	19.2	1/2	3
			9.6	1/2	5

*Assigned to links of nominal quality.

1. Multiple Access Controller

The most significant difference between MAC five- and nine-slot software is in the part of the assignment algorithm that determines a platform's local TDMA slot availability. (This function is described in detail for the nine-slot frame in Appendix G.) The difference is due to the amount of time allotted per basic data slot in the two frame structures (five-slot = 427 ms, nine-slot = 225 ms), and how these slot durations relate to the 240-280 ms range of possible round-trip propagation delays. As shown in Fig. G-2, and as reflected in inequalities (a)-(f) presented in Appendix G and enforced by subroutine XSLT, transmissions in a given slot "N" of the nine-slot frame occur at the same time as receptions in slots N-1 and N-2. Conversely, receptions in slot N occur simultaneously with transmissions in slots N+1 and N+2. Constructing similar diagrams for the five-slot frame would show transmissions in slot N to coincide with receptions in slots N and N-1; receptions in slot N will coincide with transmissions in slot N and N+1. Consequently, inequalities (a)-(f) of Appendix G must be replaced with the following five inequalities:

$$(a) \quad T_N + I_N + R_{N-1} + I_{N-1} < H + 2F$$

$$(b) \quad R_N + T_N + I_N < H + 2F$$

$$(c) \quad R_N + I_N + T_{N+1} + I_{N+1} < H + 2F$$

$$(d) \quad R_N + I_N < H + F$$

$$(e) \quad T_N + I_N < H + F$$

Again, the subset of inequalities to be satisfied depends on whether the prospective call requires the platform in question to operate in an interactive, receive-only, or transmit-only mode. Since the "transceiver analysis" functions to be performed by subroutines XSUM and XSLT for the five-slot frame are, as shown above, significantly different from those for the nine-slot frame, two independent versions of MAC software have been developed for the different frame structures.

Small differences exist between the five- and nine-slot MAC real-time structures. Although the real-time events RS0, R0, ..., R4 are unchanged, they occur at slightly different times relative to the start of frame pulse. Figure 2-2 shows the timing for the nine-slot MAC. The times are changed for the five-slot MAC due to (a) the different amount of control burst and system status data output by the MAC and (b) the different frame duration.

Finally, various MAC data base components, parameters, tables, and I/O buffers are structured differently in the five- and nine-slot versions. The most significant of these differences are (relative to the nine-slot MAC):

(a) The assignment list is 36 blocks (144 words) smaller, having to describe only 45 TDMA slots.

(b) The satellite capacity maps only describe 45 TDMA slots (though occupying the same number of computer words).

(c) The control burst output buffer is longer by two parity-check blocks.

(d) The System Operator's Console (system status data) output buffer is shorter by 288 bytes.

(e) The "link adaption table", indicating the burst and code rates to be used for given composite worst-case RFI and signal-to-noise ratios, is altered to reflect the data in Table D-1.

(f) The Table describing nominal arrival times in the frame for ranging and report/request bursts (upon which satellite range corrections are based) is altered to reflect the different positions of the ranging and request/report subslots in the five-slot frame structure.

2. Call Simulator

The five- and nine-slot versions of the Call Simulator software differ only in the values of a few constants, including: frame duration, number of basic

	SLOTS				
	0	1	2	3	4
CH 1				NET-011 112-113	
CH 2	CONTROL	FED REP	NET-014		PR-5 NET-017 106-021
CH 3		PR-5 NET-002			
CH 4					
CH 5					
CH 6		PR-4 NET-013 112-114	PR-2 NET-014 112-115		
CH 7			PREEMPT		
CH 8				CONTROL	FED REP
CH 9					

TACS CIRCUIT ASSIGNMENTS

1. TACS CIRCUIT ASSIGNMENTS ARE SUBJECT TO CHANGE WITHOUT NOTICE.

data slots per frame, number of request/report subslots per basic data slot, number of bytes in the control burst, and all of the request/report burst arrival times. Both Call Simulator versions use the same real-time software structure, since processing time is not a critical issue and the frame durations are comparable.

3. System Operator's Console

A single version of the System Operator's Console software is used for both nine- and five-slot system operations. As only one version exists, most of the five/nine-slot differences are explicitly noted in Section IV. Only one of the console's displays, the "frame structure" display, differs in format during five-slot operations. The five-slot frame structure display is presented in Fig. D-1, and may be compared with Fig. 4-3.

4. Scenario Generator

As described in Section V, the Scenario Generator program must perform a transceiver analysis upon each subscriber unit that is to be added to a given net. This is done in order to verify that the subscriber unit has access to sufficient transceiver capacity to support the net. The Scenario Generator uses the same subroutines, namely XSUM and XSLT, as used by the MAC assignment algorithm for the transceiver analysis function. As explained under "Multiple Access Controller" above, very different versions of these subroutines are required by the nine-slot and five-slot frame structures. Hence, for the same reasons as for the MAC software, two independent Scenario Generator programs have been developed for the two different frame structures. Other than the transceiver analysis subroutines, there are no major differences between the five- and nine-slot Scenario Generators.

APPENDIX E
MULTIPLE ACCESS CONTROLLER DATA BASE

The operation of TACS is contingent upon the existence of a MAC data base that consistently reflects all aspects of system status that are relevant to MAC functions. The data base describes the current system configuration: the number and communications capabilities of TAC units; the number and characteristics of communications nets; the organization of the system's end users (TAC I/O ports) into nets; the satellite channels being managed; the location of an alternate control community, if any; the number, location, and types (shared, dedicated, or polled) of ranging and request/report subslots provided; etc. All elements of system configuration remain static in the current TACS implementation (except for the channels being managed, which may be altered by the system operator). Dynamic management of such items as the number, location, and type of request/report subslots in the system would be expected in a full-scale TACS. Changes in configuration would be initiated by the system operator (or by the MAC itself) in response to changes in system loading and would be reflected in the MAC's data base.

The MAC data base also dynamically describes the demand assignment activity in the system. This includes the queue of pending circuit requests, and information on the currently assigned circuits. The circuit assignment information is provided from the viewpoint of the subscriber units involved, the nets involved, and the satellite capacity used.

The major components of the MAC data base are: (1) the request queue, (2) the assignment list, (3) the TAC list, (4) the net list, and (5) the satellite capacity maps. (Note that the Call Simulator maintains its own copies of the assignment list, TAC list, and net list for use as its data base.) The data base was designed for use on 16-bit machines, with ease of access and flexibility as major considerations. To that end:

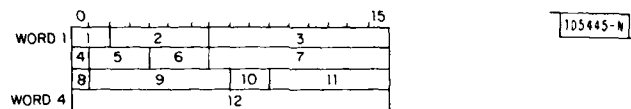
1. Some information is redundant. For example, whether or not a particular TDMA slot is presently assigned can be determined from either the assignment list or the satellite capacity maps. The latter is merely a shorthand (binary) representation.
2. All entries are explicit, with no clever coding used to "save bits." An example of such coding would be the case of two separate entries, one with three possible values and the other with five possible values. To store these entries explicitly requires five bits (fields of two and three, respectively), but the fifteen combinations could be expressed in four bits. Note also that the bits for any given entry (with the single exception of call duration in the assignment list) are always packed into a single 16-bit computer word.
3. To the greatest extent possible, the lists have uniform block size, minimizing the amount of random searching for information. In the TAC list,

for example, the information pertaining to each TAC occupies a block of exactly seven words of memory, and a block is allocated for every TAC address, regardless of whether that TAC is currently operational. The advantages of such a structure are twofold. First, given a TAC address, a pointer to the information about that TAC can be calculated, obviating a search from the top of the TAC list. Second, the TAC address need not itself appear in the list.

The following sections detail the formats of the data base components.

1. Request Queue

The request queue contains circuit requests that are awaiting service by the MAC's assignment algorithm. The data for each request occupy a block of four computer words, as shown in Fig. E-1. The blocks are placed in a first-fit buffer, with the order of the assignment algorithm's access to the blocks controlled by a pointer list. The pointer list orders the requests primarily by precedence and secondarily by their frames of arrival.



REQUEST QUEUE BLOCK

KEY TO FIELDS

- (1) SPARES
- (2) CALL DURATION
- (3) CALLING PARTY'S TAC ADDRESS
- (4) BROADCAST/TWO-WAY CIRCUIT
- (5) CALL PRECEDENCE
- (6) NUMBER OF PREVIOUS REQUEST ATTEMPTS
- (7) CALLED PARTY'S TAC ADDRESS
- (8) FLAG FOR: LATE ENTRY TO NET
OR CONFERENCE CALL
- (9) NET ADDRESS
- (10) & (11) SPARES
- (12) FRAME OF REQUEST'S ARRIVAL

ASSIGNMENT LIST BLOCK

KEY TO FIELDS

- (1) BURST RATE
- (2) CALL DURATION (Least Significant Bits)
- (3) CALLING PARTY'S TAC ADDRESS
- (4) BROADCAST/TWO-WAY CIRCUIT
- (5) CALL PRECEDENCE
- (6) NUMBER OF CONTIGUOUS SLOTS ASSIGNED
- (7) CALLED PARTY'S TAC ADDRESS
- (8) CONFERENCE CALL FLAG
- (9) NET ADDRESS
- (10) CODE RATE
- (11) CALL DURATION (Most Significant Bits)
- (12) FRAME OF ASSIGNMENT

Fig. E-1. Format of request queue block and assignment list block.

Requests initially arrive at the MAC in ten-byte blocks as shown in Fig. C-3. Appendix C gives a detailed description of this format. Processing of these blocks by the MAC's input algorithm results in a four-word block being generated for each request. The queueing algorithm places these blocks in their proper positions in queue. Comparison of the request block format of Fig. C-3 with the request queue block format of Fig. E-1 shows fields (2) - (9) to be identical. These fields specify the "request parameters" to the assign-

ment algorithm. Fields (1), (10), and (11) of Fig. E-1 are indicated as "spares" their contents have already been acted upon by the input algorithm. Field (12), containing the frame of day of the request's arrival at the MAC, is appended by the input algorithm.

2. Assignment List

The assignment list provides information on the usage of every TDMA slot in each of the nine FLEETSAT channels. A four-word block is provided for each slot. The blocks are ordered primarily by channel and secondarily by slot.

Figure E-1 depicts the format of assignment list blocks for slots which are currently assigned to communications nets. Blocks are repeated for any assignment that involves more than one contiguous slot. These blocks are directly derived from requests that have been assigned satellite capacity, hence the similarities in the assignment list block and request queue block formats. Field (3) - (5), (7) and (9) contain the same data in both formats. Field (2) in the request queue format contains the call duration specified in frames for record nets and in minutes for voice nets. Since the call duration is always specified in frames in the assignment list blocks, the six bits of field (11) as well as the five bits of field (2) are required to specify the duration of voice net assignments. Fields (1), (10), and (6), respectively, specify the burst rate, code rate, and number of slots being used by assigned circuits. Field (12) contains the frame number in which the circuit was assigned. Field (8) indicates point-to-point net circuits that are supporting conference calls, i.e., cases where net members other than the original calling and called parties have been assigned to the circuit.

TDMA slots that are not assigned to communications nets are in one of the following states: in the pool of slots available for assignment, not currently in the demand assignment pool, or in use for system overhead. The assignment list blocks for these slots are not formatted as shown in Fig. E-1, but contain a code indicating which state the slot is in. In the case of slots being used for system overhead, the codes distinguish between the types of overhead (such as slots being used for control and ranging subslots, and slots being used for primary, secondary, or tertiary sets of request/report subslots). The codes and their interpretations are shown in Fig. E-2. They are placed in the low-order byte of the "frame of assignment" field of the assignment list block, and the remainder of the block is set to zero. Note that code "10," indicating that a preemption has just occurred in a given slot, is only used in the copy of the assignment list forwarded to the System Operator's Console.

3. TAC List

The TAC list contains seven-word blocks, one describing every TAC in the system. The blocks, formatted as shown in Fig. E-3, are ordered by their respective TAC addresses. Fields (1), (5), and (8) describe a TAC unit's communications capabilities: the type of transceiver (full or half duplex) it is

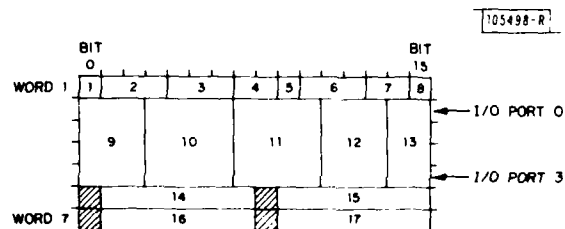
105446-N

CODE	SLOT STATUS
0	AVAILABLE FOR ASSIGNMENT
1	CONTROL AND RANGING SUBSLOTS: A*
2	CONTROL AND RANGING SUBSLOTS: B
3	PRIMARY REQUEST/REPORT SUBSLOTS: A
4	PRIMARY REQUEST/REPORT SUBSLOTS: B
5	SECONDARY REQUEST/REPORT SUBSLOTS: A
6	SECONDARY REQUEST/REPORT SUBSLOTS: B
7	TERTIARY REQUEST/REPORT SUBSLOTS: A
8	TERTIARY REQUEST/REPORT SUBSLOTS: B
9	NOT IN DEMAND ASSIGNMENT POOL
10†	PREEMPTION OCCURRED IN THIS FRAME

*"A" AND "B" REFER TO THE PRIMARY AND ALTERNATE CONTROL COMMUNITY, RESPECTIVELY

†USED ONLY IN SYSTEM STATUS DATA STREAM

Fig. E-2. Assignment list codes for slots not currently assigned to a net.



KEY TO FIELDS

- (1) FDU/HDU TRANSCIVER
- (2)&(3) UPLINK AND DOWNLINK S/N
- (4) PERCENT RFI
- (5) PRIMARY/ALTERNATE CONTROL COMMUNITY
- (6) NUMBER OF SEQUENTIAL STATUS REPORT MISSES
- (7) OFF/ON/RECEIVE-ONLY STATUS
- (8) MULTI-TRANSCIVER FLAG
- (9)-(13) ACTIVITY FOR I/O PORTS 0-3
 - (9) CALL PRECEDENCE
 - (10) CHANNEL ASSIGNED
 - (11) FIRST SLOT ASSIGNED
 - (12) NUMBER OF CONTIGUOUS SLOTS ASSIGNED
 - (13) ASSIGNMENT DIRECTION
- (14)-(17) NET ADDRESS ASSOCIATED WITH I/O PORTS 0-3, RESPECTIVELY

Fig. E-3. Format of TAC list block.

equipped with, the control community(s) it is a member of, and whether or not it is part of a "large" platform equipped with multiple TACs and associated transceivers. (The additional TACs on a large platform are represented in the TAC list as consecutive blocks with the field (8) bit set.) The TAC's communications capabilities must be considered by the MAC when attempting to assign any of its I/O ports to a circuit.

Other TAC list fields provide the MAC's assignment algorithm with information on the TAC's current status. Fields (2) - (4) contain estimates of the TAC's link quality: the uplink and the downlink signal-to-noise ratios, and the percentage of the time RFI has been present at the TAC unit's receiver. These estimates are sent to the MAC in periodic status reports from the TAC unit and are used to determine the burst rate and code rate required for any new assignments involving the unit. Field (6) contains a count of the number of consecutive status reports that the TAC has missed. If this count becomes large enough, the TAC's transmit timing error is potentially larger than that allowed for by the TDMA frame's data slot guard intervals. (The MAC uses status report information to issue satellite range updates.) The MAC is able to recognize this situation, placing the TAC in "off" status in field (7). The TAC must transmit a ranging burst to the MAC in order to regain "on" status. (Note that the uplink signal-to-noise ratio in field (2) and the "receive-only" status option in field (7) are not fully implemented in the current version of TACS.)

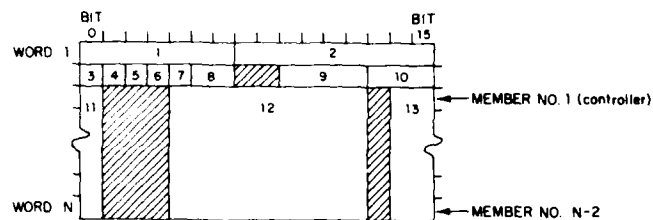
The remainder of the TAC list fields describe the status of the TAC's four I/O ports. Fields (14) - (17) give the address of the net that each of the I/O ports is associated with. (An indication is also given of any I/O port that is not "connected" to a net.) If any of the I/O ports is currently assigned to a circuit, details of the assignment are given in fields (9) - (13). Such information on the TAC unit's ongoing communications activity is necessary for the functioning of the assignment algorithm. Note that field (13) contains the assignment direction as seen from the TAC unit represented by the TAC list block in question. For example, a broadcast circuit would have an assignment direction of "transmit" in the calling TAC's block, and "receive" in the called TAC's block.

4. Net List

The net list contains information on the characteristics, membership, and status of the TACS communications nets. The format of the data block used to describe each net is given in Fig. E-4.

The number of members in a net, and hence the size of a net list block, is not uniform. Field (2), defining the block size, is used in locating net blocks deep within the list. Unlike the TAC list, a net list block is not necessarily provided for each net address and the blocks are not necessarily in order of increasing address. Field (1) is therefore used to explicitly identify the net address for a block.

105499-R



KEY TO FIELDS

- (1) NET ADDRESS
- (2) NET LIST BLOCK SIZE, N (words)
- (3) NET PRESENTLY ON/OFF THE AIR (for all-member nets)
- (4) ALL-MEMBER/POINT-TO-POINT NET
- (5) CONTROLLED/NON-CONTROLLED NET (for point-to-point nets)
OR
BROADCAST/TWO-WAY NET (for all-member nets)
- (6) RECORD/VOICE NET
- (7) NET'S CONTROL COMMUNITY (primary/alternate)
- (8) NET'S BASEBAND DATA RATE
- (9) FIRST SLOT OF SET TO WHICH NET CONSTRAINED (for all-member nets)
- (10) NUMBER OF SLOTS IN SET TO WHICH NET CONSTRAINED (all-member nets)
- (11) NET MEMBER PRESENTLY ON/OFF THE AIR (for point-to-point nets)
- (12)&(13) NET MEMBER ADDRESSES (TAC address and I/O port number)

Fig. E-4. Format of net list block.

Fields (4) - (10) define a net's characteristics: the general class of net (all-member or point-to-point, record or voice, etc.), the control community to which the net's members belong, the baseband data rate at the net members' I/O ports (a rate of 2400 bps is assumed in the current implementation), and the set of slots reserved for all-member nets. (Note that the slot set reservation mentioned in the last item refers to reserving, at each net member's transceiver, the capacity necessary to communicate in the slot set. Satellite capacity is not reserved.) Fields (12) and (13) specify the net membership, i.e., which I/O ports on which TACs belong to the net. For all-member nets and for controlled point-to-point nets, the first member listed is the net controller. Field (11) indicates the off/on status of each member of a point-to-point net; the bit in field (11) is set if the net member in question is currently assigned to a circuit. Since all-member net circuits always involve the entire net membership, the single bit of field (3) is adequate for describing the off/on status of the net.

5. Satellite Capacity Maps

The satellite capacity maps provide a shorthand representation (as compared to the assignment list) of the status of every TDMA slot in each of the nine satellite channels. Information is given on whether or not each slot is currently in use, and on the precedence level of those slots that are being used. (Each request, and hence each assignment in TACS is tagged with a precedence level ranging from 1 (highest) to 5 (lowest).) There is one map for each of the five precedence levels. Slots indicated as available for assignment in the map for precedence L are either unoccupied or are occupied by assignments of precedence lower than L. Such a slot may be assigned to any calls for precedence L or higher (after any necessary preemptions). Slots flagged as available in the precedence 5 map are unoccupied and are available for calls of any precedence. Conversely, those flagged as unavailable in the precedence 1 map are unavailable to calls for any precedence.

The layout of a segment of the satellite capacity map set is depicted in Fig. E-5. Each of the nine words in a map corresponds to a particular satellite channel, and each bit within a word corresponds to a slot within the channel (bits left justified). For example, bit position 5 from the lefthand side of the sixth word in a map corresponds to slot 5 of channel 6. When an assignment of precedence level L is made, the bits corresponding to the assigned channel and slot(s) in the map for precedence level L and in the maps for all lower precedence levels (down to level 5) are set to zero, indicating that, for those levels, the particular slots are presently unavailable. Figure E-5 indicates the representation of a precedence 4 circuit assignment in slot 4 of channel 2.

Slots that are being used for system overhead or that are not in the demand assignment pool are coded in the satellite capacity maps in the same

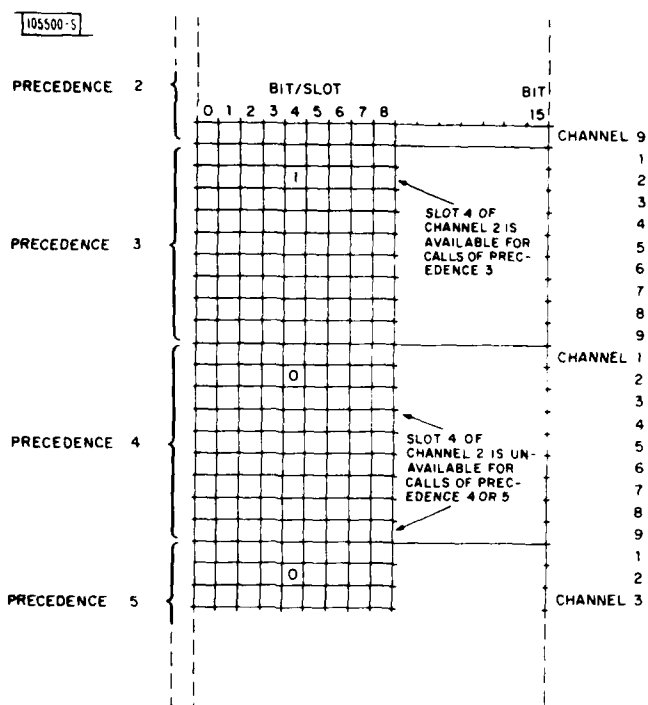


Fig. E-5. Format of segment of satellite capacity maps.

manner as precedence 1 assignments. These slots, therefore, remain unavailable for assignment to calls of any precedence until either the MAC or the system operator reconfigures the system.

Two complete sets of the satellite capacity maps are maintained, one representing the actual slot status and the other serving as a scratchpad copy for the assignment algorithm's routines. The maps are a powerful tool, as they simplify the MAC's job of searching for satellite capacity while minimizing the preemption impact of new assignments.

APPENDIX F
VARIAN MINICOMPUTER MEMORY MAP

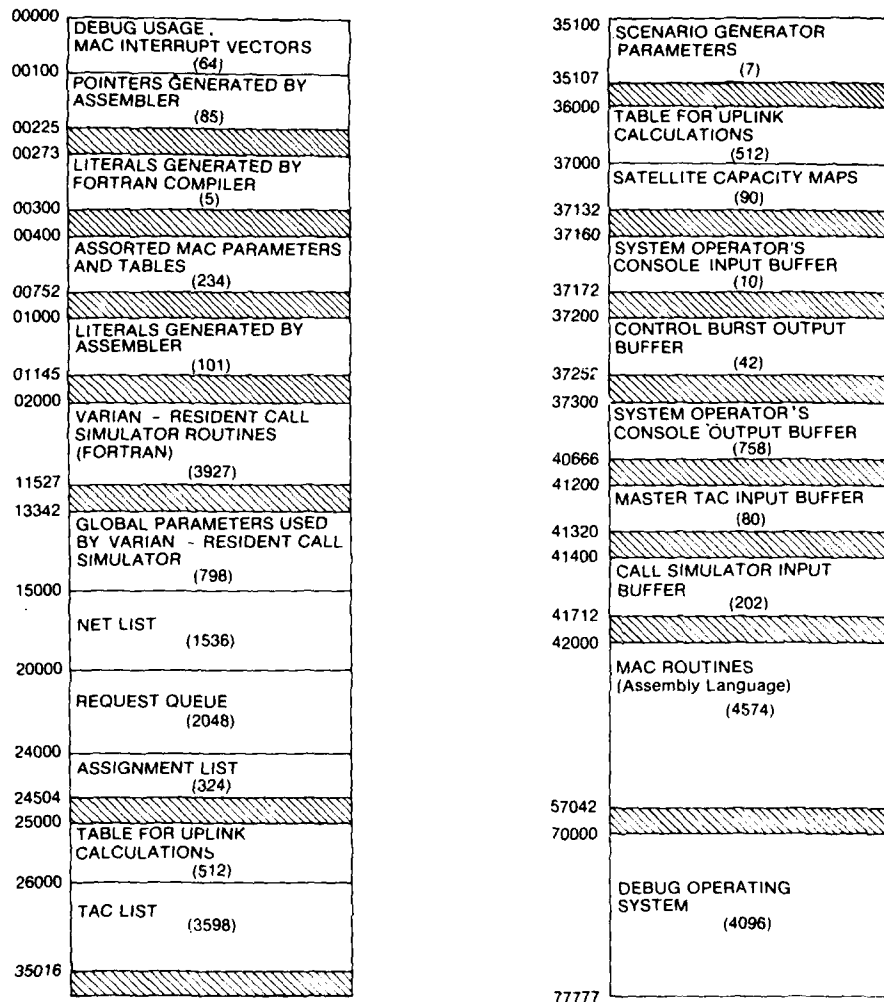
Figure F-1 is a memory map for the Varian minicomputer hosting the MAC. Each block of memory displayed in the figure contains computer instructions and/or data that are part of the MAC, the Varian-resident Call Simulator (described in Appendix H), or the Debug operating system.

The Varian 620/L-100 has 32K available 16-bit words of core memory, expressed as octal addresses 00000 through 77777. The address range occupied by each block in Fig. F-1 is indicated (in octal) to the block's left. The block size (in decimal words) is indicated beneath the block's label. The figure, clearly, is not drawn to scale.

The memory organization of Fig. F-1 is the result of a long evolutionary software development process. The sizing and positioning of the memory blocks have been influenced by many factors, including Varian hardware architecture, the structure of the Debug operating system, MAC software details, debugging convenience, etc. The reader, therefore, should not be overly concerned with the exact size and location of each item in memory; Fig. F-1 is primarily presented to provide an order of magnitude feeling for the amount of computer memory that has been required by the components of the current MAC implementation. A brief description of the memory segments displayed in Fig. F-1 follows.

The "lower" 100₈ and the "upper" 10000₈ words of Varian memory are reserved for the use of the Debug operating system (except for a few locations used by the MAC for interrupt vectors). Other low-address memory locations are occupied by the pointers and literals generated by the assembly of the MAC code. Literals are also generated by the compilation of the Varian-resident version of the Call Simulator. The Varian-resident Call Simulator, written mostly in FORTRAN, interfaces with its small number of assembly language subroutines via common access to a block of global parameters. The Varian simulator/MAC interface is similarly accomplished by allowing common access to the data base.

The remaining segments of Varian computer memory make up the MAC. The net list, request queue, assignment list, TAC list, and satellite capacity maps are the major components of the MAC data base. Other memory segments are also part of the MAC data base: the tables used to calculate the uplink quality of system subscribers, parameters describing the current scenario (besides those in the TAC and net lists) passed to the MAC by the Scenario Generator, and other assorted parameters and tables. The MAC also has three input buffers and two output buffers used for its interface with the three other Central Control Facility units. (The control burst output buffer is used for output to both the Master TAC and the (non-resident) Call Simulator.) Note that a small number of assembly language "utility" routines, used for MAC testing and for Varian-resident Call Simulator operations, are included in the memory block labeled "MAC routines".



105444-N

Fig. F-1. Varian minicomputer memory map.

APPENDIX G
MAC ALGORITHM
FOR
DETERMINATION OF LOCAL TDMA SLOT AVAILABILITY

In the course of considering a circuit request from a member of a point-to-point net, the MAC must determine which of the time slots each party to the request is capable of accessing. The question is one of transceiver (radio) availability. An algorithm which lends itself to computer implementation has been designed to make that local slot availability determination for an individual terminal. This algorithm takes into account the requirement that each terminal must have continual access to both the control and request/report subslots, and also accounts for the terminal's ongoing communications traffic of precedence greater than a specified preemption threshold. The algorithm is generalized in that it is applicable for terminals equipped with either half- or full-duplex radios, in any number.*

The slot availability algorithm consists of two stages. Stage one, accomplished by a subroutine designated XSUM, consists of accumulating information about terminal configuration and all pertinent transceiver activity, and placing that information in an organized, tabular form. Stage two, implemented in the subroutine XSLT, accepts the transceiver configuration/activity table as input, compares activity to resources, and outputs the desired binary map of the slots that the terminal is capable of using to support a new circuit. The remainder of this appendix details the two stages of the algorithm, and, in recognition of the fact that the explanation still may not be perfectly clear, concludes with an example of slot availability determination.

1. Tabulation of Local Transceiver Configuration and Activity
(Subroutine XSUM)

Inputs to this process include the complete address (i.e., TAC unit and I/O port) of the party to the call and the current value of that party's preemption threshold. The address is used to calculate a pointer to the block of information in the TAC list which pertains to the particular subscriber unit. An individual TAC list block, described in Fig. E-3, contains all available information concerning a given subscriber unit's (and through a one-to-one association, it's transceiver's) configuration and present communications involvements. Adjacent blocks in the TAC list may indicate other transceivers (and hence TAC units) which are colocated on the same platform as the terminal

*"Large platforms" are platforms equipped with multiple TACs and their associated transceivers. The transceivers on such platforms are expected to be used in a pooled fashion, i.e., the total platform transceiver capacity is assumed to be shared among all of the platform's TAC units for establishing circuits. Note that such an arrangement requires that only one transceiver per platform must access the control and request/report subslots at any given time.

105503-R

TRANSCEIVER COUNT	{		FDUX
			HDUX
	{		SLOT 0
			1
			2
			3
RECEIVE COUNT R			4
			5
			6
			7
			8
	{		0
TRANSMIT COUNT T			
	{		WORD 14-COUNT OF PLATFORM'S TRANSCIVERS INVOLVED IN TRANSMIT-ONLY CIRCUITS IN TIME SLOT 2 (considering all satellite channels and pre- emption threshold)
			7
			8
			0
			1
			2
			3
INTERACTIVE (two-way) COUNT I			4
			5
	6		
	7		
		SLOT 8	

Fig. G-1. Transceiver configuration/activity summary table (generated by XSUM).

and which might also be used to support the prospective new call. If necessary, the MAC repositions the TAC list pointer upwards such that it directs attention to the first block in the list which pertains to a transceiver at the platform in question. The MAC then makes a single downward pass through the region of the TAC list corresponding to the platform, and accumulates transceiver-oriented information in a table structured as shown in Fig. G-1. Upon completion of the information gathering process, the first two entries (words) in the table contain total counts of half- and full-duplex radios at the platform. One of these entries must be zero, since the terminals on a platform must be either totally full-duplex or totally half-duplex radio equipped. The third table entry is a count of all current receive-only activities (circuits) requiring the transceivers during the first time slot in the frame. Similarly, the rest of the table reflects receive-only, transmit-only and interactive (two-way) activities in the various slots of the TDMA frame. Continual receipt of the control subslot and ability to transmit in the request/report subslots are reflected in the table. Certain current communications activities for the terminal are not included in the table. First, any call already occupying the I/O port needed for the prospective new call is ignored, since that old call must terminate in order for the new one to commence. Second, no activity of precedence below the preemption threshold is logged into the table. Thus the table indicates terminal transceiver status after any virtual preemptions needed to free I/O or transceiver resources.

2. Generation of Slot Availability Map (Subroutine XSLT)

The desired final map consists of a single 16-bit computer word, with each one of the leftmost nine bits corresponding to one of the nine TACS TDMA slots and indicating whether or not that slot is accessible by the transceivers available to the particular user. Ones signify that the slot is available, and the map is initialized (optimistically!) to an "all-ones" state.

For each slot, individually, the MAC must compare current local transceiver activity to total local transceiver resources. This comparison may be expressed in terms of the six inequalities listed below:

- (a) $T_N + I_N + R_{N-2} + I_{N-2} < H + 2F$
- (b) $T_N + I_N + R_{N-1} + I_{N-1} < H + 2F$
- (c) $R_N + I_N + T_{N+1} + I_{N+1} < H + 2F$
- (d) $R_N + I_N + T_{N+2} + I_{N+2} < H + 2F$
- (e) $R_N + I_N < H + F$
- (f) $T_N + I_N < H + F$

The quantity T_N is the total count of transmit-only activities (circuits) requiring the platform's radio(s) during slot N, and is available directly by using N as a pointer in the T segment of the transceiver activity table (refer

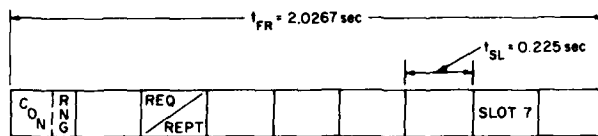
to Fig. G-1). Similarly R and I correspond to counts of receive-only and inter-activer (two-way) circuits. Subscripts must be evaluated modulo 9. The designations H and F refer, respectively, to the total numbers of half-duplex and full-duplex radios available on the platform. At this point, it is worth noting that $H + 2F$ is the maximum number of data streams which can be handled simultaneously by the platform's radios. Also, $H + F$ is the maximum number of receive or transmit operations which the platform can simultaneously support.

Due to the TDMA slotting and to propagation delays, uplink usage of a given slot impacts downlink usage of several other slots (and conversely). With reference to Fig. G-2, if a half-duplex-radio equipped terminal were transmitting in slot 7 it could not simultaneously support circuits requiring data receipt in either of slots 5 and 6. Inequalities (a), (b), and (f) relate to the uplink, and downlink capacity is characterized by the analogous inequalities (c), (d), and (e). Figure G-2(c) gives a large-scale presentation of the timing relationships for a single time slot, designed slot N. The four regions labeled "A" through "D" on the time axis relate directly to inequalities (a) through (d), above, and encompass the total time impacted by slot N. Starting with uplink considerations and more specifically examining region A, it can be seen that transmitting (T or I) in slot N occurs simultaneously with receiving (R or I) in slot N-2. This observation leads immediately to inequality (a), which indicates that in order to establish a new uplink in slot N, the total of ongoing activities must be less than the total radio capability. An identical, but time-shifted, argument applies to region B (and inequality (b)). Finally, inequality (f) expresses the constraint that, for both regions A and B, the total of existing transmit circuits must be less than the total number of radios. The origins of the three downlink inequalities follow from similar arguments.

If the prospective new call is to include interactive traffic, then the MAC must test for both uplink and downlink availability. This means that all six inequalities must be strictly satisfied. For calls requiring receive-only and transmit-only capability, respectively, the algorithm only requires satisfaction of inequality sets $\{(c), (d), (e)\}$ and $\{(a), (b), (f)\}$.

3. Example of Slot Availability Map Generation Process

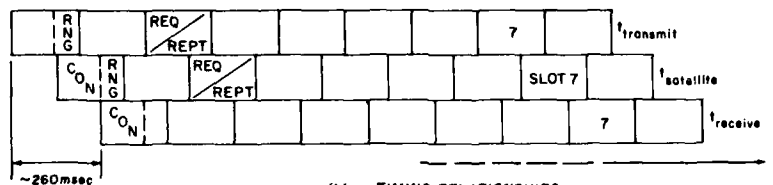
This example illustrates the algorithm by which the MAC determines in which TDMA slots a platform has the available transceiver resources with which to participate in new communications activities. For illustrative purposes, a quite simple platform configuration has been chosen, as depicted in Fig. G-3. It is assumed that the platform has a single available radio and that besides accessing the control and request/report subslots, the platform is presently involved in two communications circuits. One circuit is an interactive (two-way) voice call in slot number 3 and is labeled "V1" on the middle time axis ($t_{\text{satellite}}$) of the figure, and the other circuit is the receipt of broadcast record traffic in slot number 8 and is labeled "R1." Lower left to upper right



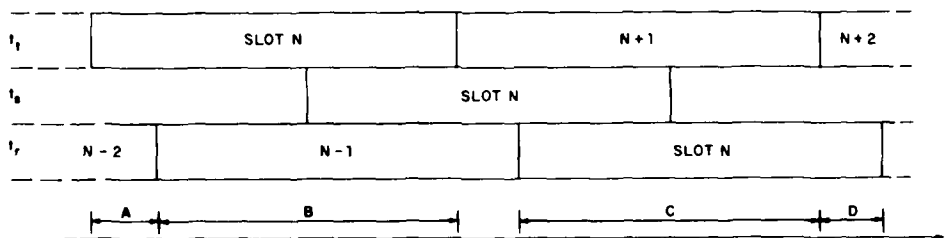
105504-R

CON: CONTROL SUBSLOT
RNG: RANGING SUBSLOTS (2)
REQ/REPT: REQUEST AND REPORT SUBSLOTS (6)

(a) CHANNEL WITH SYSTEM CONTROL

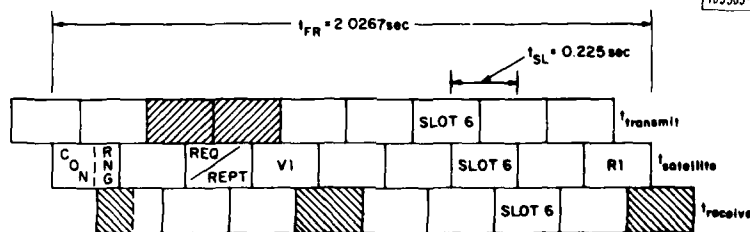


(b) TIMING RELATIONSHIPS



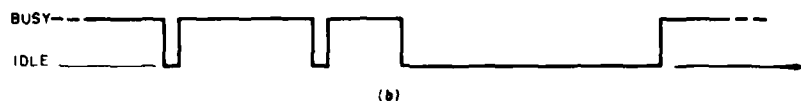
(c) DETAIL OF SLOT TIMING RELATIONSHIPS

Fig. G-2. Frame structure timing relationships.



105505-R

(a)



(b)

CON: CONTROL SUBSLOT
RNG: RANGING SUBSLOTS (2)
REQ/REP: REQUEST AND REPORT SUBSLOTS (6)
VI: TWO-WAY VOICE TRAFFIC
R1: ONE-WAY RECORD TRAFFIC

Fig. G-3. Example frame timing diagrams.

hatching indicates time intervals during which the platform's transmitter is potentially busy. Similarly, upper left to lower right hatching indicates receiver busy periods. Part (b) of Fig. G-3 depicts a transceiver idle/busy function, and is applicable only in the case of a half-duplex radio.

The first stage of the algorithm involves tabulation of ongoing transceiver activity (including control and request/report subslot access) as a function of TDMA slot for the platform. The subroutine XSUM is passed a preemption threshold, and ignores all ongoing communications which fall below the threshold; thus low precedence calls are effectively preempted. For this example, it is assumed that the preemption threshold is set at the lowest level, and hence all of the platform's ongoing traffic is included in the table (Fig. G-4).

105506-R

TRANSCIVER COUNT	FDUX	0	
	HDUX	1	
RECEIVE COUNT R	SLOT 0	1	← CONTROL SUBSLOT
	1	0	
	2	0	
	3	0	
	4	0	
	5	0	
	6	0	
	7	0	
	8	1	← CIRCUIT "R1"
TRANSMIT COUNT T	0	0	
	1	0	
	2	1	← REQUEST/REPORT SUBSLOTS
	3	0	
	4	0	
	5	0	
	6	0	
	7	0	
	8	0	
INTERACTIVE (two-way) COUNT I	0	0	
	1	0	
	2	0	
	3	1	← CIRCUIT "V1"
	4	0	
	5	0	
	6	0	
	7	0	
	SLOT 8	0	

Fig. G-4. Example transceiver configuration/activity summary table.

The second stage of the algorithm results in generation of the desired binary map of available slots. The input to that subroutine (XSLT) is the transceiver configuration/activity table, which is used to evaluate, slotwise, the set of strict inequalities previously described. If all of the necessary inequalities are satisfied for a particular TDMA slot, the platform has the transceiver resources required to work that slot. The top part of Fig. G-5 tabulates the values of the two sides of each inequality for each TDMA slot, for our example half-duplex terminal. Each entry (box) contains two numbers, separated by a diagonal line. The number above the line is the value of the left-hand side of the inequality, and conversely. Below the inequality table are the three possible output binary slot maps. The top map, labeled I for interactive or two-way circuits, contains unity entries in those slot positions for which all six inequalities ((a) through (f)) are satisfied, and hence indicates slots which could be used by the platform to support a new interactive call. In a similar vein, the middle (R) map contains unity entries in the slot positions for which inequalities (c), (d) and (e) are satisfied. Note that this map indicates exactly what is easily observable from Fig. G-3 - that the platform's receiver is available during slots 4 through 7.

105507-S		SLOT									
		0	1	2	3	4	5	6	7	8	
INEQUALITY	a	0/1	1/1	2/1	1/1	0/1	1/1	0/1	0/1	0/1	
	b	1/1	1/1	1/1	1/1	1/1	0/1	0/1	0/1	0/1	
	c	1/1	1/1	1/1	1/1	0/1	0/1	0/1	0/1	1/1	
	d	2/1	1/1	0/1	1/1	0/1	0/1	0/1	0/1	1/1	
	e	1/1	0/1	0/1	1/1	0/1	0/1	0/1	0/1	1/1	
	f	0/1	0/1	1/1	1/1	0/1	0/1	0/1	0/1	0/1	
MAPS	I	0	0	0	0	0	0	1	1	0	
	R	0	0	0	0	1	1	1	1	0	
	T	0	0	0	0	0	0	1	1	1	

Fig. G-5. Inequalities and slot maps for example half-duplex platform.

Pursuing this example one step further, let us assume that the platform has a single full-duplex radio. The timing diagram (Fig. G-3(a)) and the transceiver activity part of Fig. G-4 remain the same, but the resultant inequalities and slot maps are as depicted in Fig. G-6. The greater capability of a full-duplex radio manifests itself in the form of more ones in the maps! Again the maps indicate what is obvious from Fig. G-3 - e.g., that the full-duplex radio equipped platform can add a new interactive call using any slot from the set {1,4,5,6,7}.

105508-S

		SLOT								
		0	1	2	3	4	5	6	7	8
INEQUALITY	a	0	1	2	1	0	1	0	0	0
	b	2	2	2	2	2	2	2	2	2
	c	1	1	1	1	0	0	0	0	1
	d	2	1	0	1	0	0	0	0	1
	e	1	0	0	1	0	0	0	0	1
	f	0	1	1	1	1	1	1	1	1

MAPS	I	0	1	0	0	1	1	1	1	0
	R	0	1	1	0	1	1	1	1	0
	T	1	1	0	0	1	1	1	1	1

Fig. G-6. Inequalities and slot maps for example full-duplex platform.

APPENDIX B VARIAN CALL SIMULATOR

The TACS Call Simulator was originally programmed on the Varian minicomputer. Its first application was as a source of diverse circuit requests with which to debug the MAC's assignment algorithm software. Since the Call Simulator and MAC were running on the same machine, they needed no interrupt-driven real-time structure to maintain synchronism. On an alternating basis, each routine (MAC and simulator) took control of the Varian minicomputer and accomplished all of its required processing for a given time frame, before handing the computer over to the other routine. The total time required for combined MAC and simulator processing per frame was on the average much less than the TACS frame duration of approximately two seconds, so in that sense the MAC and Varian Call Simulator could run in faster-than-real time.

Ultimately the MAC was interfaced with the real TACS subscriber units, at which point all TACS equipment was forced to adopt real-time structures conforming to the frame structure. For two reasons, the Call Simulator was thereafter separated from the MAC, and placed on a Digital Equipment Corporation (DEC) PDP-11/03 minicomputer. MAC software was expected to grow in complexity yet it and the simulator would have only finite processing time available, as defined by the frame structure. Thus arose the distinct possibility that during rare frames of extremely heavy circuit request activity the Varian minicomputer might simply not be fast enough to complete both routines. Placing the Call Simulator (which consists mainly of comparatively slow FORTRAN code) on the DEC machine relieved a large portion of the Varian machine's processing burden. In addition, it was felt that both MAC performance testing and system demonstrations would be much more realistic and convincing if the source driving the MAC were located in a different machine.

Even after the DEC version of the Call Simulator became available, the Varian simulator was maintained and its request generation software was upgraded in parallel as features were added to the DEC simulator. In all aspects other than request generation, the DEC simulator became significantly more realistic than its Varian-based analogue. The motivation for keeping the Varian simulator was its ability to run, along with the MAC, at a high rate independent of the TACS frame structure. This allowed quick generation of various types of demand assignment performance data, with speed gains of as much as a factor of ten.

The DEC Call Simulator is described in detail in Section III of this report. In many regards, especially with respect to request generation, the DEC and Varian Call Simulators are identical, so this appendix highlights only the significant differences. The first two sections list functional and implementation differences, respectively. The final section describes the Varian Call Simulator's algorithm for transmitting requests to the MAC.

1. Functional Differences

The Call Simulator available on the Varian minicomputer differs from the final, realistic, PDP-11/03-based Call Simulator in the ways that it handles the following functions:

a. Status Report Generation

The Varian Call Simulator generates no periodic status reports from simulated active terminals. In addition, the Varian simulator does not offer dedicated subslots for transmission of either reports or requests. Thus all requests must use random access (shared) subslots, and they are all subject to contention. Another ramification of not generating reports is that no subsequent range update messages must be sent in the control burst. As a result, an unrealistically large amount of the control burst capacity is available for demand assignment messages (i.e., assignments, preemptions and call progress messages) issued by the MAC.

b. Contention for Shared Request/Report Subslots

Rather than sending all requests to the MAC, waiting, and assuming contention in the event of no prompt response, the Varian Call Simulator itself figures out which of its requests would have contended and it only passes on non-contending requests to the MAC. All requests which collide wait for one frame only and are then immediately retransmitted in a randomly chosen request/report subslot. Naturally the requests are once again subject to contention and further retransmission. The Varian Call Simulator does not use the "number of previous request attempts" field embedded in the request format. Instead, the last word of a request generated by the Varian simulator and residing in the MAC's queue is the frame of the request's generation (not the frame of arrival at the MAC; see Fig. E-1). By comparing current frame count to frame of generation, the MAC (or person operating the MAC) can determine how many times a request was transmitted before successfully getting through.

c. Request Queueing

When the MAC is operating with the Varian Call Simulator, its input algorithm, which transfers incoming requests from the Call Simulator (DMA) input buffer to a "temporary" buffer preparatory to queueing, is disabled. The Varian Call Simulator places its requests directly into the temporary buffer. In essence, the part of the MAC which is necessary for operation with the real subscriber units is short-circuited, so it is impossible to drive the MAC simultaneously with real and Varian-simulated requests.

1. Clocking for Multiple Pending Requests

The TACS subscriber units are programmed not to allow a given user (i.e., a particular TAC's I/O port) to make a new circuit request if the user already has a request pending. The DEC version of the Call Simulator also follows this protocol, by comparing the calling party address in each newly generated request to the calling party address in all requests being held awaiting response, and casting out the newly generated request if it constitutes a "duplicate." The Varian Call Simulator does not perform such a test. Hence one must be careful in interpreting MAC performance test data generated with the Varian simulator. The second of a pair of pending requests to arrive at the MAC will normally find that the first request has received an assignment. The result is both unrealistic and does not conform to the input process for most queueing models; we have a particular user either blocking (if the second request is of equal or lower precedence than the first) or preempting himself.

2. Implementation Differences

The Varian and DEC Call Simulators differ in some ways which do not affect functional performance, but which are significant enough to warrant mention:

a. Real-time Operation

As previously indicated, the Varian Call Simulator does not require any real-time software structure to maintain synchronism with the MAC. The two routines may be operated together at a very high rate of speed. If, however, it is desired to use the display/command package of the System Operator's Console, the MAC must conform to the standard TACS two-second frames. It is possible to invoke MAC real-time mode of operation, and the Varian Call Simulator simply "comes along for the ride." Note that there is only one version of the Varian Call Simulator software, whereas the DEC Call Simulator with its real-time software exists in both five- and nine-slot frame structure versions.

b. Data Base

A Call Simulator needs an up-to-date version of part of the MAC's data base. Since it is resident in the same machine, the Varian simulator simply accesses the data base lists which are maintained by the MAC. The DEC simulator, on the other hand, includes a large body of software dedicated to reflecting current system status in its own data base lists.

c. Random Number Generator

The DEC Call Simulator uses the FORTRAN-invokable random number generator function which is provided with Digital's RT-11 operating

system. The Varian simulator calls a general-purpose random number generating subroutine which was written at Lincoln Laboratory for a project previous to TACS.

d. Number of Request/Report Subslots

The number of request/report subslots available to the DEC Call Simulator is one, two, or three times the number which will pack into a basic data slot, the total number being dependent on whether the MAC operator has opened up any secondary or tertiary request/report subslots. The DEC simulator extracts that information from the MAC's control burst. The Varian Call Simulator may work with any number of (shared) request/report subslots in the range one to sixteen. The exact number must be specified at program initialization time by the MAC/simulator operator.

e. Request Holding Limit

The Varian Call Simulator has a request buffer limit of 100 requests, including those generated in earlier frames being held for retransmission and those generated in its current frame. When the buffer becomes filled, the simulator stops generating new requests (for the current frame) and also increments a counter. At the end of a simulation run, the operator may inspect the counter to determine during how many frames the buffer was filled. In practice, the buffer capacity was never taxed.

3. Request Transmission Algorithm

The Varian Call Simulator's request transmission scheme takes over immediately after generation of the current frame's new requests is completed. Both held-over and new requests are expected to be packed at the top of the simulator's request buffer (designated IHOBUFF in the software). A 16-element array (designated KTHSLT), each element of which corresponds to a particular request/report subslot, must be initialized to an all-zero state. The request transmission algorithm consists of the following steps:

- a. Sequentially, for each request in the IHOBUFF buffer (other than those requests emanating from the central control site, and readily identifiable by the fact that their calling TAC address is in the range 1-9) randomly chose a request/report subslot. If the position in KTHSLT corresponding to that subslot is still zero, the subslot has not been used by any other request. "Capture" that subslot by replacing the zero with a pointer to the request. If the position in KTHSLT corresponding to the chosen subslot is a positive number (i.e., pointer to another request which has tried to claim use of the subslot) then a collision has just occurred. Replace the pointer with a minus one to indicate contention. Finally, if the position in KTHSLT corresponding to the chosen subslot is a minus one, then two

or more requests have already collided in that slot. No change in the contents of the KTHSLT array is necessary.

b. Proceed sequentially through the KTHSLT array, looking for positive, non-zero entries. These are pointers to requests whose transmissions were successful (i.e., did not contend with any other requests). Use the pointer to locate the request in IHOBUFF. Move the request to the MAC's temporary (queueing) buffer, and zero out the old copy of the request in IHOBUFF.

c. Sequentially from the top, search IHOBUFF for requests emanating from the central control site. Move any such requests found into the MAC's temporary buffer and zero out the old copy in IHOBUFF.

d. At this point, IHOBUFF consists solely of requests which contended with each other for request/report subslot capacity, and of randomly spaced blocks of zeros where the requests which were successful in getting through to the MAC used to be. Consolidate all the remaining requests into the top part of request buffer. New requests which are generated in the next frame, will be placed immediately below the held-over requests.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ESD-TR-81-1	2. GOVT ACCESSION NO. AD-A102 928	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) TACS Central Control Facility	5. TYPE OF REPORT & PERIOD COVERED Technical Report	
		6. PERFORMING ORG. REPORT NUMBER Technical Report 542
7. AUTHOR Steven B. Storch Susan N. Landon	Lee E. Taylor Carole D. Cappello	8. CONTRACT OR GRANT NUMBER(s) F19628-80-C-0002
9. PERFORMING ORGANIZATION NAME AND ADDRESS Lincoln Laboratory, M.I.T. P.O. Box 73 Lexington, MA 02173		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Program Element No. 33109N
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Electronics Systems Command Department of the Navy Washington, DC 20360		12. REPORT DATE 12 February 1981
		13. NUMBER OF PAGES 190
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Electronic Systems Division Hanscom AFB Bedford, MA 01731		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES None		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) TACS time division multiple access automated demand assignment techniques		
Multiple Access Controller Call Simulator		
System Operator's Console Scenario Generator		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The Terminal Access Control System (TACS) is designed to make efficient usage of 25-kHz-wide satellite transponder channels, such as those available on FLEETSat, GAPSAT, and LEASAT. It does so by applying time division multiple access and automated demand assignment techniques to these channels, allowing a significant increase in both data throughput and supportable user population, as compared to the initial, single network per transponder FLEETSat operating mode. The Terminal Technology Group of Lincoln Laboratory Division has built prototypes of all of the TACS ground segment equipment, as well as special-purpose test equipment used to demonstrate and quantify the performance of TACS. This report describes the implementation details of components of the TACS "Central Control Facility" as configured for testing at the Laboratory. The components described are: the Multiple Access Controller, the Call Simulator, the System Operator's Console, and the Scenario Generator. All differences in implementation configuration between the testing environment and a full-scale operational environment are noted in the report.		

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DATE
FILMED
-8